Memoria Proyecto de Sistemas Informáticos:

Videojuego Educativo sobre Introducción a la Programación

Autores:

García Azpiazu, Margarita

Nicolás Fúnez, Cristina

Vaquero Curto, Alberto

Profesor director:

Natalia López Barquilla

CURSO 2008/2009

Facultad de Informática

Universidad Complutense de Madrid

Agradecimientos

A nuestros compañeros del proyecto anterior, Antonio De Miguel Vicenti y Borja Gómez Gómez, por su gran trabajo realizado.

Índice

Tabla de Figuras

Resumen:

El objetivo principal de este proyecto es la creación de un videojuego educativo dirigido a los alumnos de 1º de Ingeniería Informática, y en general a cualquier persona que se enfrente por primera vez ante un lenguaje de programación imperativo.

Se trata de una aventura gráfica ambientada en el Campus de Ciudad Universitaria de la UCM. Durante el desarrollo del juego, el jugador se irá moviendo por las distintas facultades e irá adquiriendo nuevos conocimientos de programación a medida que la trama avance.

El proyecto se ha desarrollado utilizando un software de desarrollo de Videojuegos llamado Adventure Game Studio. Esta herramienta es de uso libre y no requiere ninguna licencia.

Abstract:

The main aim in this Project is the creation of an Educational Videogame intended for first year Computer Science Students, and in general for anyone who face up for first time to any imperative Programming Language.

It is about a graphic adventure, set in the Campus of the UCM in Ciudad Universitaria. During the game development, the player will move through the different Faculties and will acquire new knowledge of Computer Programming as the plot advances.

The project has been developed using a videogame development software called Adventura Game Studio. This tool is open source and requires no license.

Palabras clave:

AGS, aventura gráfica, conceptos de programación, didáctica de programación, introducción a la programación, minijuegos, videojuego educativo.

.

1. Introducción:

Hoy en día la Informática está tan arraigada en nuestras vidas, que muchas veces no nos percatamos del enorme esfuerzo y trabajo que existe detrás.

Gran parte de los jóvenes que se enfrentan cada año a la decisión de decantarse por un futuro profesional, ven la Informática como una continuación de lo que ellos mismos hacen ya en casa: videojuegos, trabajar con aplicaciones, páginas web… Algunos de estos jóvenes, incluso deciden que en un futuro quieren ser ellos mismos los que desarrollen estos juegos o websites. En el momento en que esto ocurre, ya está, ha surgido un nuevo informático.

Lo que normalmente no saben, es cómo los que ya somos informáticos (o poco nos queda) conseguimos comunicarnos con el ordenador. Cómo darle las órdenes precisas para que cuando ellos hagan clic, éste responda adecuadamente ante lo que le han pedido. Desconocen que igual que entre las personas nos podemos comunicar y entender ya que compartimos un mismo lenguaje, con un ordenador ocurre lo mismo. Los programas y aplicaciones, ya sean: juegos, páginas Web etc.… se crean porque el informático, llamémosle programador, conoce un lenguaje de programación, es decir, un lenguaje "comprensible" por la máquina.

El lenguaje de programación se transforma mediante una aplicación en lenguaje máquina que es el verdaderamente comprensible por la máquina.

Llegados a este punto podríamos desvelar a todos estos principiantes, que cuando un programador está utilizando un lenguaje de programación para comunicarse con el ordenador, lo que esta haciendo… es precisamente programar.

Cuando hace unos meses nos plantearon realizar este proyecto, capturamos en seguida la esencia, su razón de ser. Lo mejor manera de aprender nuevos conceptos de una forma práctica y entretenida es jugando. Recordando la forma en la que nosotros comenzamos a adquirir conocimientos de programación, que fue mediante el típico libro aburrido de cientos de paginas (en el que normalmente para cuando pasabas de hoja se te había olvidado lo que habías leído en la anterior), nos dimos cuenta de que no conocíamos ningún juego, de entre los muchos que existen, que enseñara a programar, ninguno que facilitara la labor de aprender a programar a futuros alumnos, aprendiendo conceptos nuevos mediante ejemplos prácticos y además de una forma amena y divertida.

Nuestra labor no consistía en crear un juego desde cero, sino ampliar y mejorar uno ya existente desarrollado por 2 compañeros de la Facultad de Informática como Proyecto de Sistemas Informáticos en el curso 2007-2008.

Dicho juego, podía considerarse un producto acabado puesto que constaba ya de un inicio, una trama coherente y un final que cerraba la historia perfectamente. El trabajo que se nos proponía, por tanto, no iba encaminado a modificar el curso de los acontecimientos de la historia, personajes, etc. Lo que debíamos hacer era detectar problemas, errores y aspectos mejorables para conseguir que la aventura gráfica se convirtiese en un producto apetecible para futuros usuarios y, en especial, alumnos de Informática.

Además, este juego iba encaminado a que los nuevos alumnos de la Facultad de Informática de la Universidad Complutense de Madrid pudieran conocer y familiarizarse con el campus de Ciudad Universitaria, lugar donde se desarrolla íntegramente la trama de la aventura gráfica.

Así pues, este juego está pensado para aquellas personas a las que les gusta el mundo de los ordenadores, destinadas a comunicarse con ellos y que en un futuro no muy lejano, se convertirán en informáticos. Es un juego dedicado a todas esas personas a las que les gusta mucho los videojuegos, pero que hasta ahora no conocían ninguno que les mostrara cómo funcionan por dentro. Hay que aclarar que en especial, los más beneficiados por este proyecto serán los futuros estudiantes de informática de la Universidad Complutense de Madrid, ya que no sólo les entretiene mientras aprenden a programar, sino que además les permitirá conocer la zona en la que van a pasar algunos de los años de su vida.

Esperamos que nuestro proyecto os ayude.

Requisitos del juego:

Pentium o procesador superior.
64 Mb RAM.
Windows 98, ME, 2000, XP o Vista con DirectX 5 o superior.
Soporta todas las tarjetas de sonido y gráficas soportadas por DirectX 5.
Códecs de audio y video.

El resto de la memoria presenta la siguiente estructura: en el siguiente capítulo presentamos las principales herramientas utilizadas para desarrollar el Proyecto. En

el capítulo 3, explicamos la metodología que hemos seguido durante el proceso de desarrollo: tecnologías de las que nos hemos servido para comunicarnos entre nosotros, análisis del código existente, toma de decisiones… En cuanto a la implementación del juego, exponemos detalladamente cómo ha sido llevada a cabo en el capítulo 4, incluyendo modificaciones de aspectos ya existentes e introducción de otros nuevos. En el capítulo 5, se puede encontrar una guía del usuario que servirá como ayuda adicional en aquellos casos en los que no se sepa qué camino tomar en el juego. En el capítulo 6, describimos qué conceptos de programación pueden ser aprendidos con este proyecto. Los objetivos cumplidos con el desarrollo de este producto pueden encontrarse en el punto 7. El capítulo 8, cierra este documento con las conclusiones obtenidas a lo largo del proceso de desarrollo.

2. Herramientas utilizadas:

2.1. AGS (Adventure Game Studio)

AGS, que son las siglas de Adventure Game Studio, es un entorno de desarrollo muy fácil de usar creada por Chris Jones para la implementación y el desarrollo de juegos para PC.

Es la herramienta software principal utilizada para desarrollar el juego. Se barajaron otras posibilidades para la ampliación del proyecto, pero debido a que AGS era fácil de usar y que cambiar de herramienta supondría trasladar todo lo existente sin saber a ciencia cierta si sería viable, decidimos continuar con ella. Otra de las razones que ayudaron a continuar con AGS fue que no cuesta nada, ya que es de libre distribución. Incluso podemos crear juegos comerciales con él (sujeto a los términos de licencia).

AGS permite la creación de juegos de aventuras de una forma sencilla y eficaz.

Los juegos de aventura, aventura gráfica o point n' click poseen la mayoría una estructura bastante conocida que consiste en: se tiene uno o varios personajes que pueden visitar distintos lugares interactuando con el ambiente, ya sea obteniendo cosas, combinándolas con otras o con el mismo ambiente, y hablando con otros personajes que se encuentren allí. Las cosas que se obtengan se guardan en un lugar especial llamado "inventario", que en nuestro juego será una mochila, y el objetivo del jugador es ir resolviendo pruebas o acertijos definidos por el diseñador para poder pasar a la siguiente prueba o acertijo hasta llegar al final del juego. El

juego posee una historia que pone al jugador en un contexto que le permita resolver dichas pruebas.

La herramienta AGS contiene un editor y un compilador. El editor toma los recursos que el creador produzca, léase: gráficos, sonidos y guiones, y con ello fabrica un programa. El compilador se incluye dentro de ese programa, y es el que permite distribuir el juego a todo aquél que quiera jugarlo, sin necesidad de incluir el editor. Es decir, nosotros los desarrolladores del juego tenemos el código fuente que genera el ejecutable, siendo ambos independientes.

Las versiones del Adventure Game Studio que hemos utilizado, han sido varias:

→ AGS 2.7.1 y 2.7.2, AGS 3.0.2, AGS 3.1, AGS 3.1.1.

Siendo esta última versión la que finalmente más se ha utilizado para desarrollar el juego, ya que al abrir el código con esta versión te inhabilitaba para poder utilizar las anteriores.

En la dirección Web indicada en La Bibliografía, [7] podrán descargarse, distintas versiones del AGS.

En la Figura 2.1 mostramos cómo es el entorno del AGS en el desarrollo de un videojuego. Como se puede apreciar, tiene una interfaz simple y sencilla de utilizar.

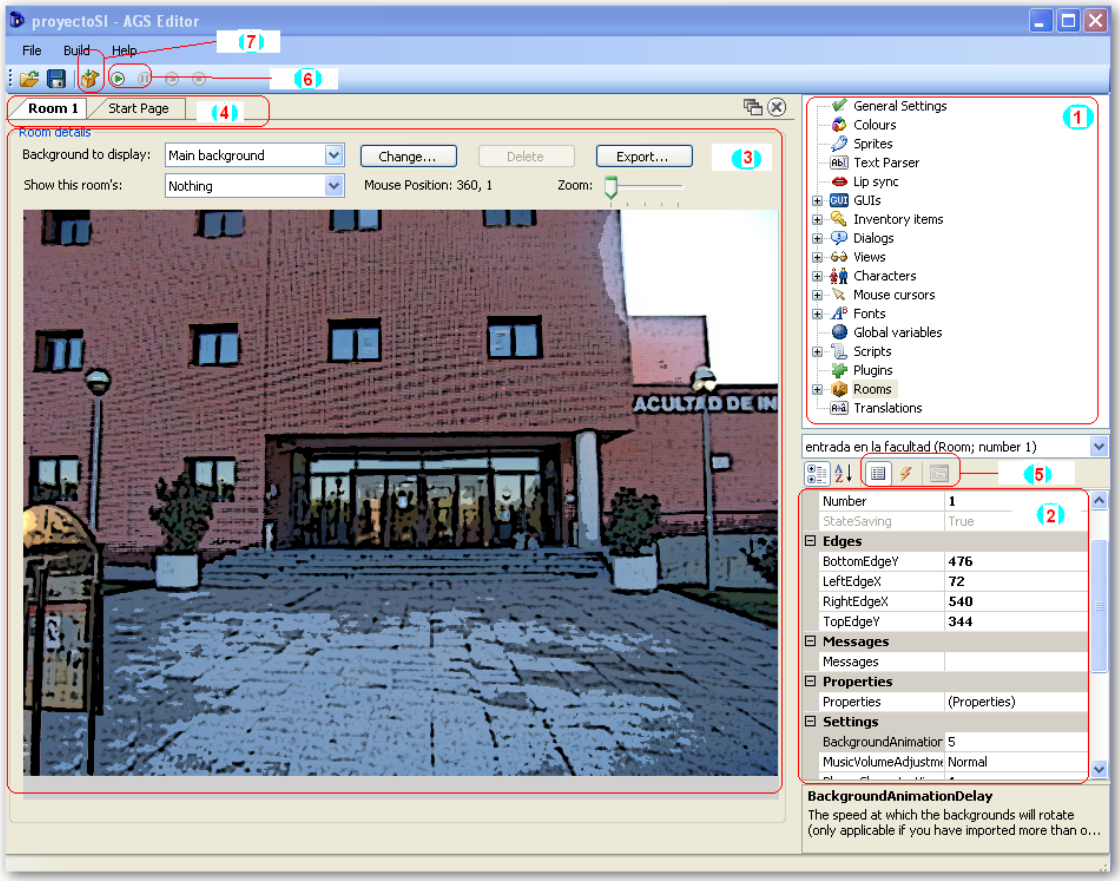


Figura 2.1: Entorno AGS

Existen determinadas zonas marcadas con una línea roja y un número, señalando las principales áreas de la ventana de trabajo de AGS. Dichas zonas hacen referencia a:

- 1. Árbol del proyecto, desde él podemos acceder a las distintas partes del juego. Da acceso a cada tipo de objeto con el que podemos trabajar.
- 2. Paleta de propiedades del objeto en particular que tengamos seleccionado en ese momento.
- 3. Zona de trabajo o ventana abierta: la parte de la ventana en la que puedes realizar los ajustes precisos sobre el elemento del árbol del proyecto que tengas abierto en ese momento.

- 4. Conjunto de fichas que muestran las ventanas que se tienen abiertas. Dichas ventanas siempre son seleccionadas desde el árbol.
- 5. Esos dos botones permiten cambiar la vista del apartado 2, pinchando sobre el rayo en lugar de aparecer la paleta de propiedades del objeto seleccionado, aparecerá una lista con los posibles eventos que atañen al objeto en cuestión, por si queremos definir alguna acción concreta.
- 6. Botón de ejecución.
- 7. Crea el ejecutable del juego.

Dentro de AGS nos encontraremos con las siguientes secciones:

→ General Settings.

Contiene las propiedades generales del juego que se está creando, como por ejemplo el nombre, la resolución, etc.…

→ Sprites.

Son las imágenes u objetos que se utilizarán dentro del juego, y que deben ser importadas al mismo, ya sea para utilizarlos como objetos, objetos de inventario o personajes.

Para almacenarlos de una forma lógica y ordenada nos permiten crear un sistema de carpetas. En nuestro caso tenemos innumerables sprites: los objetos del inventario, las distintas vistas que tendremos de un objeto... irán comentándose posteriormente según se vayan utilizando en las distintas partes del juego.

→ GUI´s

Los GUI ó Interfaces Gráficos de Usuario son regiones dibujadas sobre el fondo del juego, que contienen normalmente botones que permiten realizar determinadas tareas (guardar, cerrar, etc.…) o etiquetas que muestran determinada información como una breve definición del objeto sobre el que nos encontramos o la puntuación del juego. Otros posibles elementos que se pueden añadir en los GUI además de botones y etiquetas, son cajas de texto y listas desplegables.

Existen tres tipos de GUI:

- Los que están siempre visibles. Ejemplo: Statusline.
- Los que aparecen cuando se pone el cursor sobre la zona que ocupan. Ejemplo: Iconbar
- Y los que aparecen o se muestran ante acciones del usuario. Ejemplo: gAyuda y gInventory

→ Inventory Item

Cuando hablamos de los Inventory Item, nos estamos refiriendo a determinados objetos que el jugador puede obtener a lo largo del juego, ya sea porque aparecen en las habitaciones o porque se los proporciona algún otro personaje. Estos objetos pasan al inventario, que en nuestro juego se trata de la mochila, de donde pueden ser recuperados en todo momento por el usuario.

Más adelante se muestra la lista detallada de objetos de inventario que aparecen en nuestro juego.

→ Dialogs

Nos referimos a las conversaciones que se pueden producir entre el personaje principal y otros personajes, entre personajes secundarios, o entre personajes y objetos o determinadas áreas de una habitación. Suelen dar mayor realismo al juego, aportando dinamismo al desarrollo del mismo.

En algunas aventuras gráficas de este estilo, los diálogos eran estáticos, es decir, las frases eran siempre las mismas, pero con el AGS disponemos de árboles de diálogos completos a través de los cuales puedes elegir entre distintas opciones o temas sobre los que hablar con los personajes.

→ Views

Las vistas consisten en un conjunto de imágenes asociadas bien a un personaje o a un objeto. Lo podemos denominar como un conjunto de sprites que representan a un solo personaje u objeto, cada uno de los cuales representarán al personaje u objeto en un momento dado.

Se suelen utilizar principalmente para que se perciban mejor los movimientos de los personajes, consiguiendo así más realismo en sus movimientos, aunque otra utilidad muy práctica es la de hacer que un mismo objeto lo podamos ver con una imagen distinta dependiendo de su estado. Un ejemplo sería el libro de programación que según vas avanzando en el juego te muestra más páginas. En realidad todas las paginas del libro constituyen un único objeto y dependiendo de aquella que quieras consultar en un momento determinado se mostrará la vista correspondiente, es decir, la imagen perteneciente a la colección que le corresponda.

→ Characters

En este apartado se pueden crear, modificar y borrar los personajes que intervienen en el juego, pudiendo por ejemplo controlar cómo van a ser sus movimientos, cuándo aparecerán por primera vez, etc.…

→ Mouse Cursors

Son los distintos tipos de cursores o punteros que puede tener el ratón. Dependiendo de la acción que queremos realizar tendremos que poner el ratón en un modo u otro.

Los principales estilos de punteros para el ratón que tenemos son los siguientes:

- Mode 0 ó Walk to, su función es la de trasladar al jugador hasta la posición deseada.
- Mode 1 ó Look at, nos permite observar determinadas partes de la habitación u objetos.
- Mode 2 ó Interact, facilita la captura de objetos principalmente para añadirlos al inventario.
- Mode 3 ó Talk to, sirve para comenzar una conversación o diálogo.
- Mode 6 ó Pointer. Es el puntero del ratón habitual de Windows, lo utilizamos por ejemplo para seleccionar una de las posibles opciones de respuesta en los diálogos que aparecen durante el juego.
- Mode 7 ó Wait. Es el modo espera utilizado cuando el juego esta procesando algo que hace que el jugador tenga que esperar.

→ **Fonts**

Hacemos referencia a los distintos tipos de letra que se pueden utilizar durante el juego. Antes de que aparezcan han de ser añadidos en esta sección. De este modo, se puede utilizar un tipo de letra distinto para los diálogos y para mostrar la descripción de un objeto.

→ **Scripts**

Normalmente aquí nos podemos encontrar dos archivos de vital importancia para el diseño del juego, ya que contienen las variables, funciones y procedimientos que son globales, es decir, que son comunes o se pueden invocar desde cualquier parte del juego simplemente utilizando su nombre. Estos archivos son:

- GlobalSCript.ASH. Es un archivo donde se tienen que declarar las variables de ámbito global, es decir, accesibles desde cualquier parte del juego. Ejemplo: import int MapaH;
- GlobalScript.ASC. Es el archivo en el que se declaran variables, funciones y procedimientos de ámbito global, es decir, contiene las líneas de código que son accesibles desde cualquier parte del desarrollo del juego.

En AGS existen dos partes principales en las que se suele definir código: una de estas partes está formada por estos dos archivos donde se definen todas las líneas de código que sean globales, la otra parte es particular de cada habitación desde la que se puede acceder a las variables, funciones y procedimientos definidos en los archivos GlobalScript.asc y GlobalScript.ash.

Por ejemplo, si necesitamos utilizar una variable de estado durante el juego que me indique qué páginas del libro de programación puedo ver y cuándo, o una

variable que me diga los puntos que tengo, se pueden definir aquí e irle asignando los valores dentro de los archivos propios de cada habitación o en el mismo GlobalScript.asc.

→ Rooms

Se trata de una de las piezas angulares de este tipo de programas ya que en ellas se van a ir desarrollando las acciones del juego y será donde el jugador interaccione directamente con los escenarios o los objetos incluidos en ellos.

Las habitaciones tienen un fondo que suele ser una imagen importada. Sobre ese fondo o imagen se van a ir definiendo las zonas de la habitación y los objetos que se pueden añadir a las mismas. Las zonas que se pueden definir en una habitación son las siguientes:

- EDGES o Bordes. Representados mediante cuatro líneas, dos horizontales y otras dos verticales que normalmente indican la zona de salida de la habitación para el jugador.
- OBJECTS u objetos. Se trata de los objetos o imágenes añadidas al fondo de la habitación con los que el jugador puede interactuar. Estos objetos deben de ser importados de los Sprites existentes.
- HOTSPOT ó Puntos Calientes. Son determinadas partes de la habitación con cierta importancia y que normalmente comprenden una especie de objeto contenido en la imagen de fondo con las que el jugador puede interactuar.
- WALKABLES AREAS ó Areas Transitables. Son las zonas de la imagen de fondo que define la habitación por las que el jugador puede caminar.

- WALKABLES BEHINDS. Son partes de la imagen de la habitación por las que se supone que al pasar el personaje sobre ellas deben ocultarle, con ello se consigue una sensación de realismo.

- REGIONS ó Regiones. Son partes especiales de la habitación en las que al usuario, al trasladarse sobre ellas, desencadena alguna acción previamente asignada.

Las habitaciones suelen disponer de dos vistas:

→ En la primera podemos definir todos los aspectos de diseño, ya sea la imagen de fondo, las zonas del escenario, las propiedades tanto de la room como de las regiones, objetos, etc.…

→ La otra vista es en la que se muestra el código local, es decir, las variables, funciones y procedimientos propios de la room y que normalmente utilizamos para atender a los eventos que se pueden producir sobre la propia habitación o alguna de las partes de la misma.

Con esta exposición sobre AGS se pretende mostrar las partes más importantes de las que se compone, sin entrar en más detalle, es decir, no pretendemos crear un tutorial que enseñe a utilizar el AGS, sino un manual de referencia para que si quienes consulten el diseño del juego puedan entenderlo. Para aprender a manejar el AGS o consultar algunos conceptos de forma más amplia proponemos que se consulten los puntos 5, 7, 8 y 10 de la Bibliografía.

2.2. Photoshop

Para el tratamiento digital de las imágenes utilizadas en la aventura gráfica recurrimos al Photoshop. Adobe Photoshop® es una aplicación en forma de taller de pintura y fotografía que trabaja sobre un *"lienzo"* y que está destinado para la edición, retoque fotográfico y pintura a base de imágenes de mapas de bits.

Aunque el propósito principal de Photoshop es la edición fotográfica, este programa también puede ser usado para crear imágenes, efectos, gráficos y más en muy buena calidad

Aplicación en el Proyecto:

Todos los escenarios en los que se desarrolla la trama del juego son fotografías de diferentes zonas del campus de Ciudad Universitaria. Con objeto de lograr un entorno más amable para el usuario, se decidió tratar digitalmente las imágenes dándole un efecto similar al de un cómic.

2.3. Paint

Microsoft Paint está incluido en Microsoft Windows y es un programa simple para editar gráficos. Es una aplicación para procesar archivos de mapas de bits, JPEG, GIF, PNG y TIFF.

Aplicación en el Proyecto:

Todas las rooms nuevas introducidas y algunas modificadas del año pasado referentes a minijuegos y aclaraciones de conceptos, han sido realizadas o editas con Paint. Para ello, las hemos manejado como archivos PNG, ya que es la extensión que AGS reconoce.

3. Metodología:

3.1. Tecnologías utilizadas

Aquí queremos hacer referencia a otras herramientas en las que nos hemos apoyado y que nos han ayudado a conseguir nuestro objetivo final. La principal función de las herramientas que se van a nombrar a continuación ha sido la de coordinar a los integrantes del grupo, permitiendo el intercambio de datos e información.

Estas herramientas han sido principalmente el FTP y un Grupo de Google.

3.1.1. FTP:

Hemos utilizado concretamente dos Ftp en los que cada uno de los integrantes disponía de una carpeta personal accesible por el resto. En el Ftp íbamos guardando las nuevas versiones del juego con las ampliaciones que nos habíamos propuesto realizar, además de programas, archivos y paginas Web principalmente con información sobre AGS. Por lo tanto podemos decir que el Ftp básicamente nos permitía a los integrantes del grupo coordinarnos entre nosotros.

Intentamos encontrar la forma de trabajar con un solo proyecto alojado en el Ftp y realizar todas las modificaciones sobre él de forma simultánea, algo parecido a lo que hace la herramienta *Subversión de Eclipse*, principalmente para JAVA, pero no conseguimos encontrar ninguna herramienta software que nos permitiera hacerlo con el AGS y su lenguaje de programación propio.

La principal ventaja que nos daba el Ftp es que nos permitía intercambiar código sin la necesidad de quedar personalmente, con lo que nos ahorrábamos tiempo, y además nos permitía poder tener copias de seguridad y consultar versiones anteriores del código por si algo funcionaba mal en la última.

El inconveniente es que sólo existía una versión válida sobre la que se iban realizando las ampliaciones. Dicho de otra forma, en cada momento sólo podía estar trabajando uno nosotros directamente sobre el código. De modo que incluía en él sus ampliaciones y lo subía al Ftp, informando vía correo electrónico de que el siguiente integrante del proyecto ya podía descargarse la nueva versión y consultar o añadir su parte para volverla a cargar y así sucesivamente.

Como hemos indicado anteriormente, al inicio se utilizaron dos Ftp´s, ambos alojados en el mismo servidor, debido a que el primero se saturaba y comenzó a dar problemas al cargar archivos.

Para el acceso y manejo de estos dos Ftp´s se utilizaron dos programas:

- Filezilla 3.2
- CuteFtp 8, de GLOBALScape

3.1.2. Grupo de Google:

Se trata de una de las herramientas que facilita el famoso buscador de Internet. Concretamente nosotros lo hemos utilizado para enviar mensajes y entrar en hilos de debates, además de compartir algunos archivos que quedaban allí almacenados y se podían descargar en cualquier momento.

Por lo tanto también se trata de una herramienta utilizada para facilitar la coordinación entre los integrantes del proyecto, ya sean desarrolladores o tutores del mismo. Y debido a su accesibilidad, sólo es necesario disponer de Internet para hacer uso de él, por lo que nos pareció de gran utilidad.

El nombre del grupo es *PROYECTO08/09* y fue creado al principio del desarrollo con el fin de coordinarnos. Este tipo de grupos pueden ser públicos y privados, siendo el nuestro de esta segunda modalidad para evitar que personas que no pertenezcan al mismo pudieran acceder a su información.

3.2. Análisis del código existente

Se trata de una de las primeras y más importantes fases de la elaboración de nuestro proyecto. Ya que nuestro trabajo era corregir y ampliar un proyecto anterior, la primera tarea, una vez decidido que la herramienta de desarrollo que íbamos a seguir utilizando era AGS, fue entender el diseño y el código ya existentes. Para esta labor no sólo consultamos el código, que además nos sirvió para aprender más cosas sobre el AGS, sino que también nos guiamos por la memoria que habían realizado nuestros compañeros (véase Bibliografía, punto 10)

El análisis del código existente no sólo era necesario para corregir errores, sino que debíamos controlar a fondo el código, principalmente para realizar las ampliaciones de tal forma que cuando tuviéramos que ensamblar algo a lo ya existente, no tuviéramos problemas para saber donde hacerlo sin romper el orden lógico del juego.

Una de las principales labores en este sentido era entender cómo y cuándo se conseguía que los personajes actuaran, dijeran o pudieran acceder a determinados sitios y a cuáles no. En el análisis del juego vimos determinadas variables que los controlaban, lo que denominamos variables de estado, cuyo valor hacía que el personaje tuviera un determinado comportamiento o rol. El estado de cada personaje u objeto, que viene determinado por este tipo de variables, está especificado en la memoria facilitada por el anterior grupo del proyecto (véase Bibliografía, 10)

Pero no sólo debíamos comprender el sentido de esas variables. Aunque con el aprendizaje del AGS comprendimos que las habitaciones tiene su propio archivo de código, en el que se contempla todo lo referente a las mismas, la misión principal radicaba en el análisis del código global. Dicho código se encontraba en los archivos "GlobalScript" del AGS, donde se definen las variables de estado tratadas antes, otras variables de ámbito global a todo el juego, las funciones y procedimientos que debían ser accesibles desde cualquier parte, los "Parameters" que controlan las acciones a realizar después de un dialogo invocados desde ellos con el "run-script" etc.…

Podemos decir que no fue una labor sencilla, pero nos acostumbramos a consultar el código de las habitaciones que queríamos tratar y a entender el código general de los "GlobalScript".

3.3. Toma de decisiones

Al enfrentarnos por vez primera con el proyecto, el problema principal residía en decidir qué partes se dejarían igual, cuáles se modificarían y qué novedades introduciríamos en el juego.

Para introducirnos en el contexto y saber en qué fallaba y qué partes serían mejorables, durante un tiempo nos dedicamos a jugar a la aventura gráfica anotando aquellos puntos que considerábamos perfectibles.

Así pues, durante el primer mes nos dedicamos a elaborar dos listas de 'objetivos' a cumplir; una sobre las modificaciones en el proyecto existente y otra sobre las novedades que se iban a introducir.

3.3.1. Modificaciones sobre el proyecto existente:

3.3.1.1. Transformación de fotografías en imágenes caricaturizadas.

Dadas las características del AGS, esto suponía exportar todas las Rooms del juego, y tratarlas una por una con el Photoshop hasta conseguir el efecto deseado.

Una vez transformadas, debíamos importarlas de nuevo al AGS. En la mayoría de los casos, el AGS detectaba cambios en el tamaño de las imágenes por lo que no nos permitía utilizar los ajustes definidos para las fotografías iniciales.

Esto supuso volver a crear todos los hotspots, regiones, objetos, etc. además de unirlos a sus correspondientes fragmentos de código.

3.3.1.2. Modificación de algunos minijuegos:

La aventura gráfica consta, a lo largo de su trama, de una serie de minijuegos destinados a proporcionar al jugador unos conocimientos básicos de programación.

En realidad, la esencia del proyecto son estos minijuegos, por lo que consideramos que algunos de ellos debían ser mejorados.

Tal es el caso de los juegos relacionados con conceptos tales como:

- Constantes
- Variables
- Expresiones
- Asignaciones
- Tipos
- If
- While
- For
- Repaso final

3.3.1.3.Modificación del Gran Libro de la Programación:

Al modificar algunos de los minijuegos e introducir nuevos conceptos, el Gran Libro de la Programación debía actualizarse en consecuencia.

Se añadieron nuevas páginas y se tradujeron las existentes al inglés (novedad de la que hablaremos más adelante).

3.3.2. Novedades introducidas en el proyecto:

3.3.2.1. Traducción al inglés:

Se trata de un videojuego educativo dirigido especialmente a alumnos universitarios. Siendo el ambiente universitario cada vez más internacional, no podíamos sino pensar que el juego debía proporcionar la posibilidad de jugar en otro idioma, en este caso, el inglés.

Más adelante profundizaremos en este tema, para explicar detalladamente cómo se llevó a cabo.

3.3.2.2. Introducción de nuevos minijuegos:

Además de modificar algunos de los minijuegos existentes, queríamos ampliar el rango de conocimientos explicados durante el desarrollo del juego. Para ello debíamos crear nuevos minijuegos e introducirlos en la trama de la aventura gráfica.

En la versión anterior se echaban de menos algunos conceptos importantes y necesarios como los siguientes:

- Tipos enumerados
- If anidados, Tipo subrango, Case
- Punteros
- Listas, pilas y colas
- Funciones y procedimientos

3.3.2.3.Mapas

Aunque las aventuras gráficas se caracterizan por ese grado de 'dificultad' a la hora de moverse por los escenarios y encontrar los objetos y personajes necesarios para la resolución de los problemas, en este caso nos dimos cuenta de que la desorientación era demasiado grande. Por ello nos planteamos la posibilidad de incluir algún tipo de mapa que ayudase al jugador a desplazarse por los escenarios y conocer en todo momento en qué zona se encuentra.

Este fue uno de los primeros objetivos que nos marcamos, además de uno de los más importantes y urgentes junto con la ayuda y la inclusión de nuevos minijuegos.

3.3.2.4.Fases

Aunque internamente el juego estaba implementado por fases bien diferenciadas, el usuario en ningún momento tenía conocimiento ni percepción de dichas etapas.

Consideramos importante que, cada vez que aprendiese un nuevo concepto y antes de pasar al siguiente, el jugador supiese que había superado una fase y se le diese la opción de salvar la partida, salir o continuar jugando.

De esta forma además agilizábamos la dinámica del juego, haciéndolo más entretenido y paliando la sensación de aburrimiento que pudiera surgir en un juego demasiado largo.

3.3.2.5.Ayuda

Como ya hemos comentado anteriormente, las aventuras gráficas suelen caracterizarse por un cierto nivel de dificultad, que suele venir dado por los escenarios y los acertijos/problemas a resolver durante la trama del juego. Esto es un arma de doble filo, ya que tratado con cuidado puede intrigar al usuario animándole a seguir jugando o, si la dificultad es demasiado elevada, hacer que se aburra y decida no continuar intentándolo.

En este caso, consideramos que algunas de las pruebas no eran obvias y requerían mucho 'prueba y error' hasta que se daba con la combinación correcta (tal es el caso de la fregona y el aparato de aire acondicionado, por ejemplo). Por esta razón, decidimos incluir una serie de pistas en forma de ayuda, que proporcionasen al jugador el empuje necesario para seguir intentándolo en los casos en que se quedase estancado. Pensamos que la mejor forma de introducirla era por 'puntos'. A medida que el usuario interacciona con objetos, escenarios o pasa determinados niveles, se le van otorgando una serie de puntos que van abriendo diferentes ayudas. En todo momento será decisión del jugador si quiere hacer uso de las pistas o no.

3.4. Trabajo realizado sobre el proyecto existente

Nos reunimos con la profesora directora de este proyecto a principios de este curso. De esta manera, conseguimos planificar el trabajo a desarrollar en tres grandes etapas (aquí aun no sabíamos el tiempo que nos iba a llevar cada una) y marcarnos ciertos objetivos, a corto y a largo plazo.

Las tres etapas fueron: una primera, de estudio del trabajo anterior realizado y familiarización con la herramienta principal de diseño del videojuego (AGS), que duró aproximadamente un mes y medio; una segunda, de desarrollo del trabajo propuesto de duración cinco meses y medio, y una última de pruebas, de depuración de errores y de realización de la memoria, que nos llevó dos meses.

A continuación, pasamos a detallarlas:

- **Primera etapa**:

Nos reunimos con la profesora en tres ocasiones dentro de esta fase. Los objetivos de esta etapa fueron entender el funcionamiento del trabajo del año anterior (véase Bibliografía, 10), para así poder modificarlo y ampliarlo y a la vez, ser capaces de manejar con cierta soltura AGS.

Por un lado se estudió minuciosamente el trabajo realizado el año anterior en el videojuego. Nos apoyamos en la memoria correspondiente para una comprensión total de las funciones, variables, estados, etc. que se usaban en el código.

Por otro lado, necesitábamos entender el funcionamiento básico de AGS. Para ello utilizamos la memoria del año anterior. En ella, venían en detalle ciertas funciones de gran relevancia en la ejecución y desarrollo del juego. También usamos una serie de páginas web (véase Bibliografía, 3, 4, 5, 6, 7). Una de ellas, nos sirvió de gran utilidad, ya que se explicaba paso a paso cómo generar un juego corto y sencillo, pero básico, que ayudó en gran medida al objetivo marcado el principio.

Una vez superada esta etapa, pasamos a la segunda.

- **Segunda etapa**:

Comenzó con una reunión con la profesora donde especificamos, a grandes rasgos, el trabajo a realizar. Vamos a enumerar los puntos más importantes:

- Conceptos de programación nuevos a introducir.
- Conceptos de programación a modificar.
- Errores detectados para corregir.
- Aspectos para facilitar el juego.
- Aspectos estéticos.
- Novedades propuestas por nosotros.

Quedando fijados estos cometidos, hicimos un reparto de tareas inicial, con las cosas que consideramos más importantes.

Pasamos a reunirnos de forma periódica con la profesora cada viernes. De esta manera supervisaba y guiaba nuestro trabajo. Aparte, nosotros nos comunicábamos y sincronizábamos los avances en el trabajo, a través del FTP creado y de correos electrónicos.

A medida que fuimos cumpliendo los objetivos marcados en el reparto y conociendo más a fondo la herramienta AGS, y más concretamente sus limitaciones, se nos fueron ocurriendo nuevas ideas. Éstas, podrían contribuir a mejorar el videojuego. Todas ellas fueron puestas en conocimiento de la profesora y volvimos a hacer otro reparto.

Merece la pena mencionar que los objetivos marcados inicialmente se fueron refinando conforme avanzábamos en las modificaciones del videojuego, ya que al manejar AGS de una forma más profunda, nos dimos cuenta de lo que se podía hacer y lo que no y de que la manera de hacerlo podía ser más sencilla o más compleja. Debido a esto, tomamos las decisiones correspondientes con respecto a cómo hacer lo propuesto.

- **Tercera etapa**:

Cuando tuvimos el juego prácticamente terminado nos dedicamos, tanto la profesora como nosotros, a hacer pruebas para poder depurar los posibles errores. A la par, comenzamos a planificar la memoria. Hicimos un índice aproximado de la misma, el cual debatimos y fijamos con la profesora. Una vez tuvimos el índice definitivo, repartimos los puntos que lo formaban. Nos pusimos a trabajar en ellos, comunicándonos a través de un grupo de Google que creamos. Dimos a la profesora un pequeño borrador sobre los puntos más importantes. Ella nos corrigió y continuamos con el trabajo hasta tener una versión más definitiva.

Finalmente, entregamos a la profesora el borrador, tanto de la memoria, como del videojuego.

4. Implementación del juego

4.1. Minijuegos

Visto el trabajo del año anterior, nos dimos cuenta de que teníamos que cambiar minijuegos, ya que estos podían llevar a confusiones a la hora de entender algunos conceptos. También decidimos crear minijuegos nuevos para todas las ampliaciones que queríamos introducir, lo que conllevó modificaciones y creaciones de diálogos.

A continuación, pasamos a ver estas modificaciones y ampliaciones, indicando dentro de cada concepto, si ha sido modificado, o si es nuevo:

4.1.1. Constantes

Este concepto ha sido modificado.

Versión anterior: mediante diálogos se pedía a Lucas que buscara algo que pudiese ser una constante. La pista que se le daba era que ese objeto no hubiese cambiado en mucho tiempo. El objeto en cuestión, era un colgante, que pertenecía a la abuela de otro personaje del juego. De ahí, que no hubiese sufrido cambios en muchos años.

Errores o problemas: nos dimos cuenta de que esta explicación podía llevar a malentendidos. Para alguien que sabe programar, se podría explicar de esta manera. Sin embargo, para una persona que no sepa nada, puede entender erróneamente el significado de este concepto. Por ejemplo, si a ese colgante, ahora le cambias la

cadena, ¿sigue siendo una constante? No ha cambiado en mucho tiempo, pero ¿qué te impide que no lo puedas cambiar ahora?

Versión actual: intentamos transmitir la idea de que una constante es una posición de memoria donde se va a guardar un dato, el cual no se va a poder modificar (Figura 4.1).

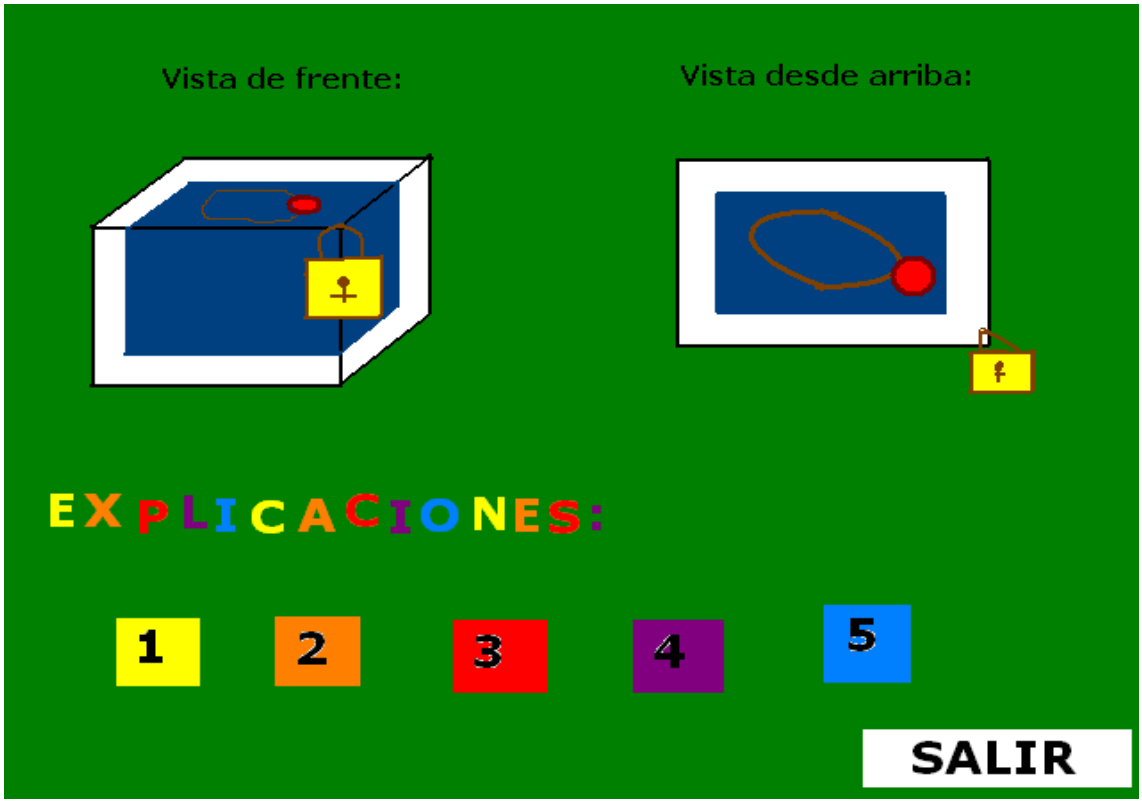


Figura 4.1: Juego constantes

Podemos observar que la posición de memoria se representa con la caja y la idea de que se va a guardar un dato se ve por el colgante que tiene dentro. Dicho colgante no se va a poder cambiar, ya que la caja tiene un candado, que no permitirá abrirla. Tan solo se puede consultar su contenido, mirando a través de ella, puesto que es transparente.

Facultad de Informática

Dentro de la caja vemos una especie de collar. Lo pusimos como pista en el juego, porque decidimos dejar tal cual el argumento principal, que se corresponde a lo contado en el resumen de la versión anterior. Aunque no hemos modificado esta idea, si la hemos matizado, como se puede ver.

A continuación, pasamos a detallar las explicaciones que aparecen con cada botón que es pinchado:

1 : **1.- DEFINICIÓN:**

"Mira el dibujo...

Una CONSTANTE la puedes imaginar como una caja transparente que no se puede abrir. Solo esta permitido ver su contenido."

2 : **2.- EJEMPLO 1:**

"Por ejemplo, podría contener un adorno de la época egipcia y que no ha cambiado en todos estos años."

"O un colgante muy antiguo... (como en el dibujo)"

3 : **3.- EJEMPLO 2:**

"También podría contener al numero PI : 3,141592654 y en un programa podríamos usarla para realizar operaciones con el numero PI."

"Se definiría constante real PI = 3,141592654;

donde PI: sería la CAJA y 3,141592654: su CONTENIDO (el adorno o el colgante)"

"En el punto 4 puedes ver esto claramente."

Facultad de Informática

4 : **4.- EJEMPLO 3:**

"constante real PI = 3,141592654;

a) Multiplicar numero PI por 2 -> PI * 2

b) Sumar 4,35 al numero PI -> PI + 4,35

c) Restar 1,8 al numero PI -> PI - 1,8"

5 : **5.- CONCLUSIONES:**

"Si en el punto 4 no hubiésemos declarado la constante PI, tendríamos que haber escrito 3,141592654 en lugar de simplemente: PI.

Como ves esto es mucho más rápido y cómodo…"

Para superar esta pantalla el jugador debe haber leído los cinco puntos, ya que el botón SALIR no permitirá abandonar este minijuego hasta no haber pulsado los cinco botones anteriormente descritos.

Pensamos, que al salir de este juego, Lucas ya estará preparado para buscar el colgante y tendrá una idea más acertada de lo que es una constante.

4.1.2. Variables

Este concepto ha sido modificado.

Versión anterior: igual que en el apartado anterior, se le pedía a Lucas que buscara algo que pudiese ser una variable, mediante diálogos. Se le daba como pista, que una variable es algo que puede cambiar y tenía que elegir entre el agua, la luna y el pelo de su madre. La respuesta correcta era el agua, ya que tiene tres estados. Una

vez sabía esto, debía encontrar agua helada (cubito de hielo), vapor de agua (una olla) y agua líquida (vaso de agua).

Errores o problemas: pensamos lo mismo que en el apartado anterior, que esta idea no expresa de forma clara lo que es una variable.

Versión actual: intentamos transmitir la idea de que una variable es una posición de memoria donde se va a guardar un dato, el cual va a poder ser consultado y modificado, siempre y cuando sea del mismo tipo que la variable.

Damos una definición aproximada de variable. Como se puede observar en la Figura 3.2, de nuevo incluimos una pista en el dibujo, ya que el jugador deberá buscar un vaso de agua, una olla y un cubito de hielo.

Pinchando en la flecha roja, pasaremos a la siguiente pantalla (Figura 3.3) donde el jugador deberá seleccionar las opciones mostradas en la Figura 3.4

Figura 4.2: Explicación variables

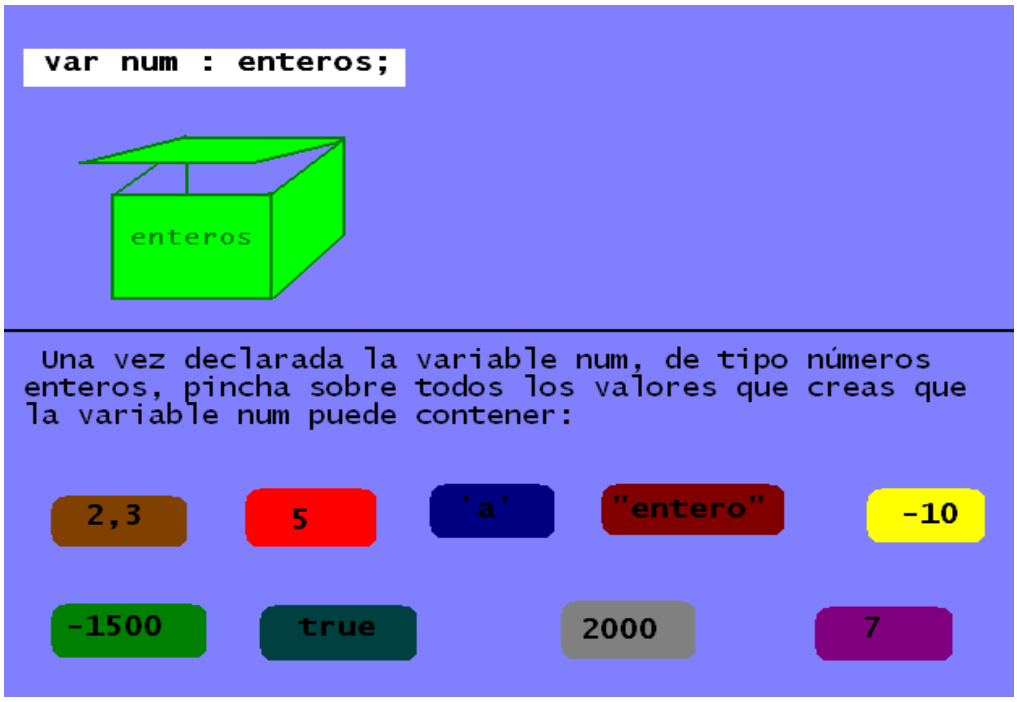


Figura 4.3: Minijuego variables

Figura 4.4: Solución minijuego variables

Creemos que tras haber pasado estas dos pantallas, tendrá una idea más clara de lo que es una variable. Estará preparado para buscar los 3 objetos requeridos, entendiendo la relación que tienen con este concepto.

4.1.3. Expresiones

Este concepto ha sido modificado.

Facultad de Informática

Versión anterior: hicieron una especie de probador de combinaciones de operaciones con ejemplos de tipos. Nos pareció una buena idea, porque el usuario podría entrenar todo el tiempo que quisiera e ir probando lo que se puede o no hacer con los tipos y así ir entendiendo como funcionan estas operaciones, la compatibilidad, las limitaciones de cada tipo, etc.

La figura 4.5 muestra la apariencia del minijuego, en el cual se mostraba el siguiente mensaje al entrar en él:

"Selecciona uno de los valores de la parte de abajo. Después un operador de la derecha, y luego de nuevo otro valor, si quieres ver otra expresión, vuelve a clickar en un valor. Si quieres salir selecciona salir"

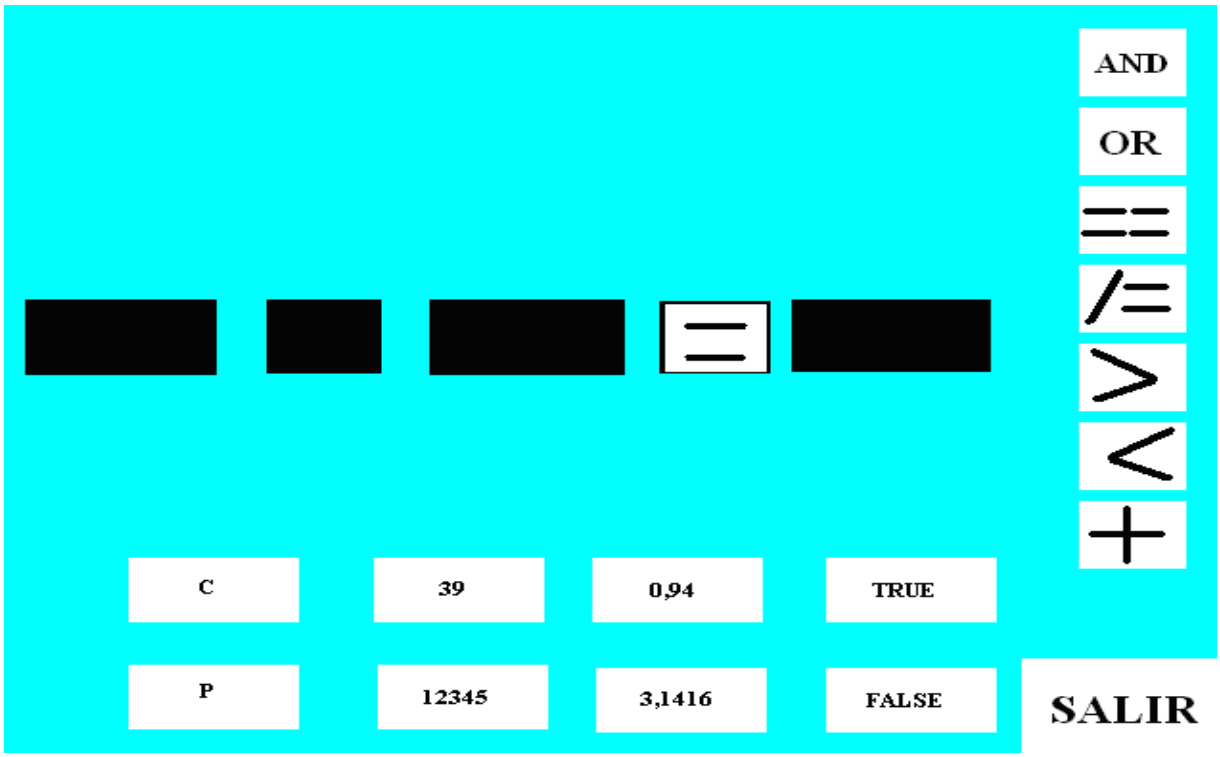


Figura 4.5: Minijuego expresiones

Al seleccionar un valor, éste se posicionaba en el rectángulo negro de más a la izquierda. Seguido a él, el operador que seleccionase el usuario y a continuación de

éste, el otro valor elegido por el jugador. Cuando estos tres huecos estaban rellenados, aparecía el resultado de forma automática en el rectángulo que está más a la derecha.

Errores o problemas: había varios errores, que pasamos a comentar:

- Aparecían los valores C y P sin las comillas simples y son de tipo CHAR. Esto, para una persona que está aprendiendo a programar, es importante.
- Había expresiones que tenían el resultado erróneo. El juego tenía errores de implementación y por ejemplo, si comparabas que 'C' fuese mayor/menor que 'P', el resultado te daba "error". Si ponías que 0,94 era mayor que 3,1416 el resultado daba "true". Así hasta que comprobamos cada caso y vimos los que estaban mal.

Versión actual: comprobamos, uno a uno, cada caso y corregimos los que estaban mal. Así, el jugador podrá probar tranquilamente todas las combinaciones que quiera y aprenderá, exactamente, lo que tiene que ser. También añadimos las comillas simples a C y P.

4.1.4. Asignaciones

Este concepto ha sido modificado.

Versión anterior: se hacía un pequeño test, mediante diálogos. Se debía responder de forma correcta a tres cuestiones seguidas. Si se fallaba una, volvía a empezar la cuenta, hasta llegar a los tres aciertos consecutivos.

Errores o problemas: la forma en que estaba programado tenía varios problemas, que pasamos a describir:

- Cuestiones implementadas mediante diálogos: éstas pasaban muy rápido, y apenas daba tiempo de leerlas. Con lo cual, resultaba demasiado complicado poder pensar la respuesta correcta. Por ejemplo, en numerosas ocasiones, apenas se podían leer los valores iniciales de las variables que luego se asignaban. De esta manera, para alguien que está aprendiendo lo que es una asignación, resultaba prácticamente imposible acertar una cuestión de este tipo.
- Test con random(): seleccionaban una de las nueve cuestiones diferentes que habían creado con un random(9). Debido a esto, en la misma partida, después de responder a una pregunta, a veces te aparecía detrás la misma cuestión. Si ocurre esto, es absurdo para el usuario responder dos veces a la misma pregunta.

Versión actual: consideramos que debíamos corregir los errores mencionados anteriormente. Pensamos que la mejor solución podía ser crear con el programa Paint tres pantallas que tuviesen tres cuestiones cada una, e ir mostrándolas de forma secuencial. Las preguntas son, o nuevas, o las que utilizaron en el anterior trabajo (algunas con modificaciones).

Una vez acertadas todas las cuestiones de la página, se da paso a la siguiente.

Lo podemos ver más claro en las figuras que vienen a continuación:

Al mostrarse cada pantalla, aparecerá la siguiente frase: "Debes pinchar con el ratón en la opción que creas correcta"

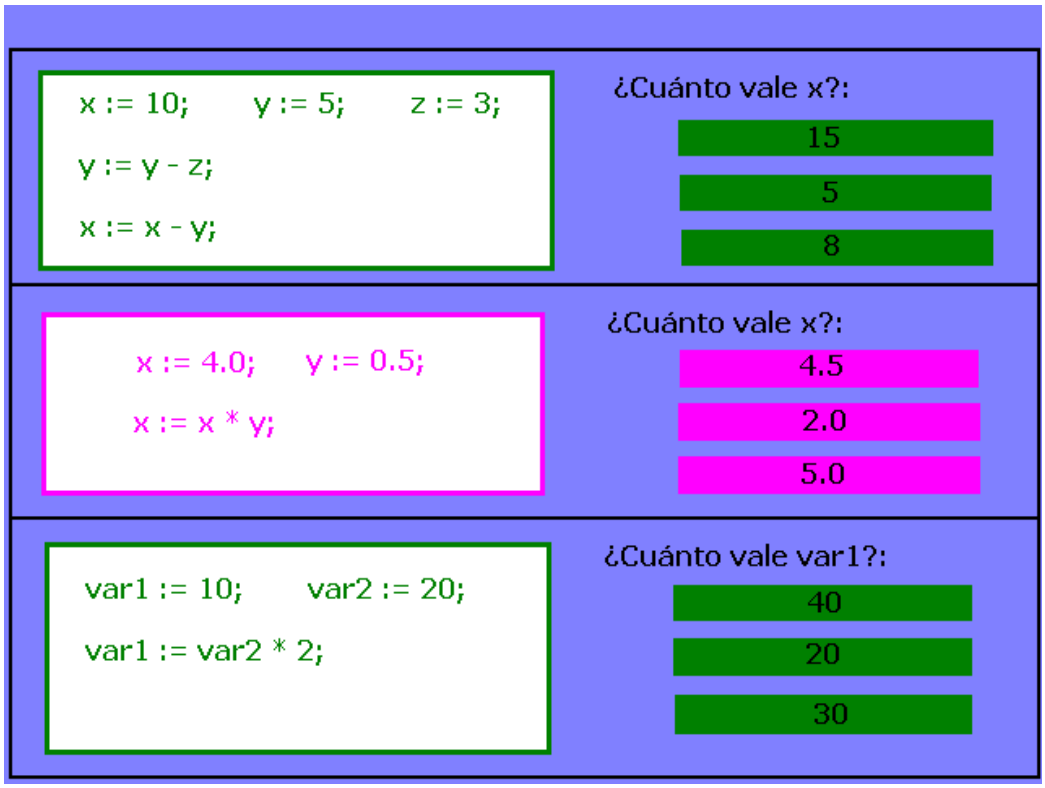


Figura 4.6: Ronda 1 preguntas asignaciones

Una vez superadas las tres cuestiones, pasamos a la siguiente pantalla:

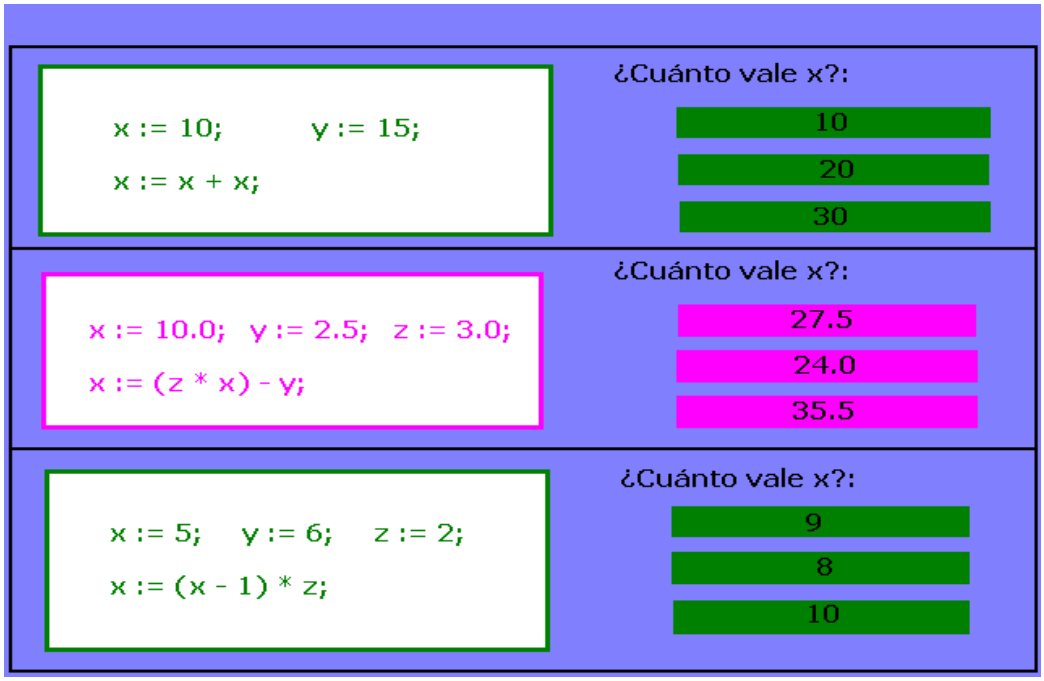


Figura 4.7: Ronda 2 preguntas asignaciones

Una vez superadas las tres cuestiones, pasamos a la última pantalla:

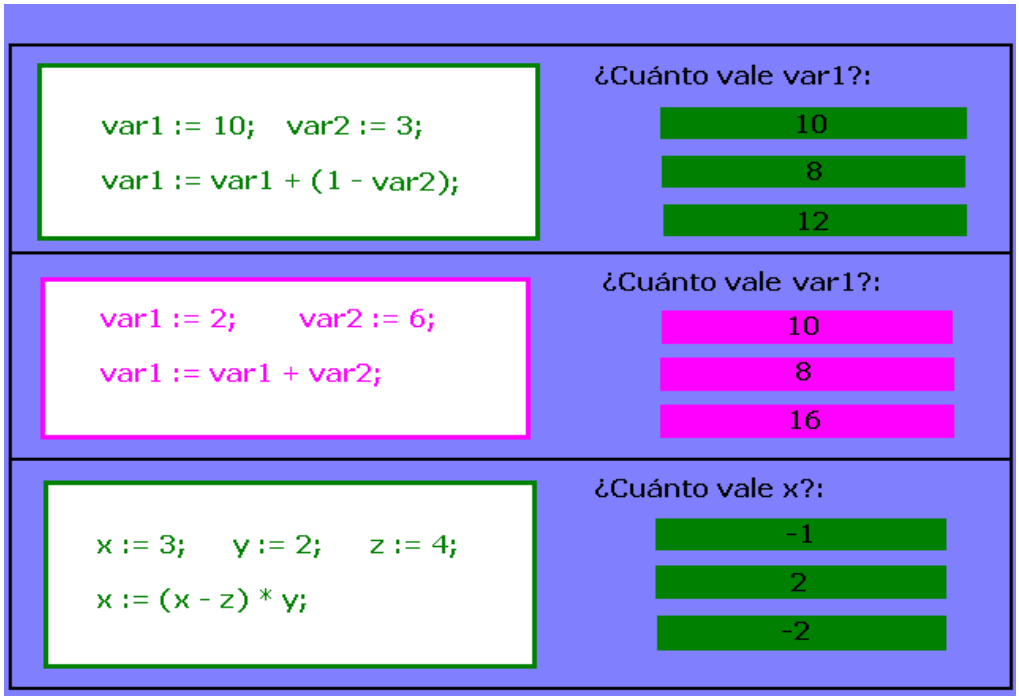


Figura 4.8: Ronda 3 preguntas asignaciones

Cuando el jugador haya acertado estas tres últimas cuestiones, habrá conseguido pasar la prueba con éxito.

4.1.5. Tipos

Este concepto ha sido modificado.

Versión anterior: hicieron un minijuego, que consistía en pinchar en el tipo de datos que quisiera el usuario y después en el ejemplo que correspondiese a ese tipo, entonces se eliminaba el ejemplo. Así hasta acabar con todos los ejemplos. Tanto los tipos, como lo ejemplos, cambiaban de posición en cada segundo (Figura 4.9)

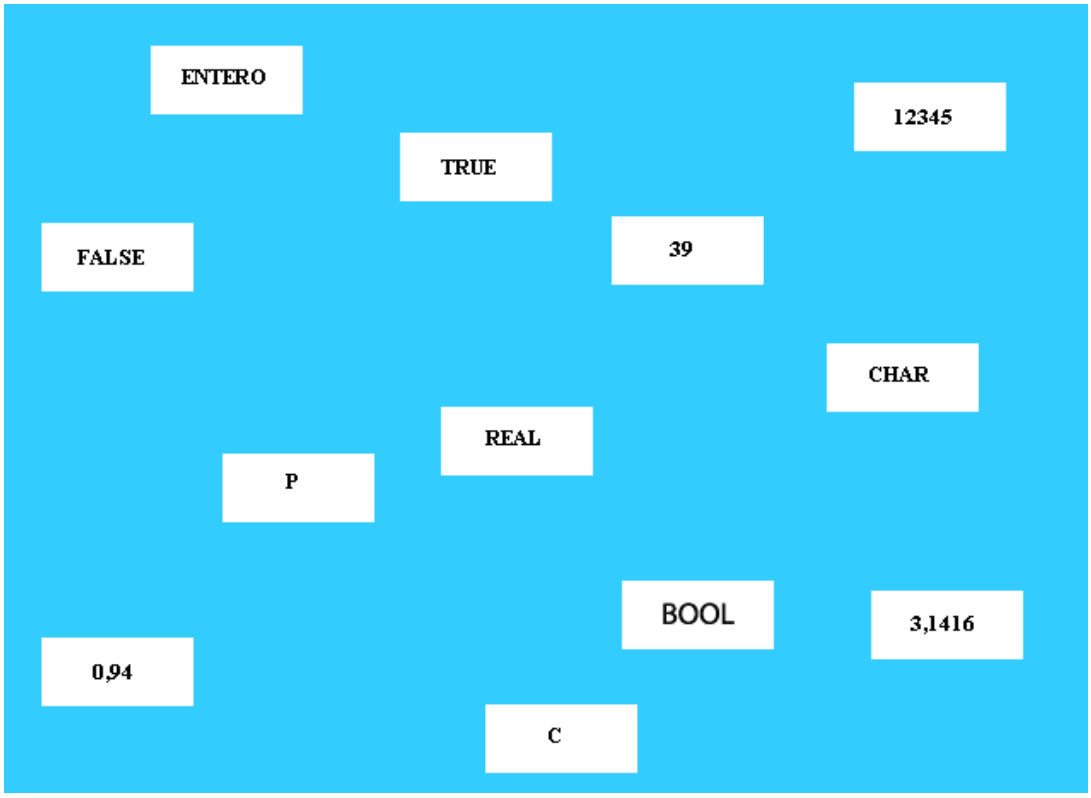


Figura 4.9: Minijuego Tipos

Errores o problemas: contenía varios errores y algunos aspectos que no nos gustaban:

- Errores: por ejemplo, las letras C y P, son de tipo CHAR, pero les faltan las comillas simples. De nuevo, consideramos este detalle importante.
- Aspectos que no nos gustaban: era excesiva la velocidad con la que los objetos cambiaban de posición. En un segundo casi no daba tiempo de verlos. Y también, a veces, los objetos se salían de la pantalla y era imposible pinchar en ellos.

Versión actual: simplemente modificamos lo comentado anteriormente, es decir, pusimos las comillas simples donde correspondía e hicimos que los objetos cambiasen cada cuatro segundos y limitamos su zona de movimiento.

4.1.6. Tipos enumerados

Este concepto lo introducimos nuevo.

Consideramos que tras explicar los tipos básicos, podríamos contar lo que son los tipos enumerados. Para ello, añadimos un diálogo introductorio tras el cual se le aparece al jugador la siguiente pantalla:

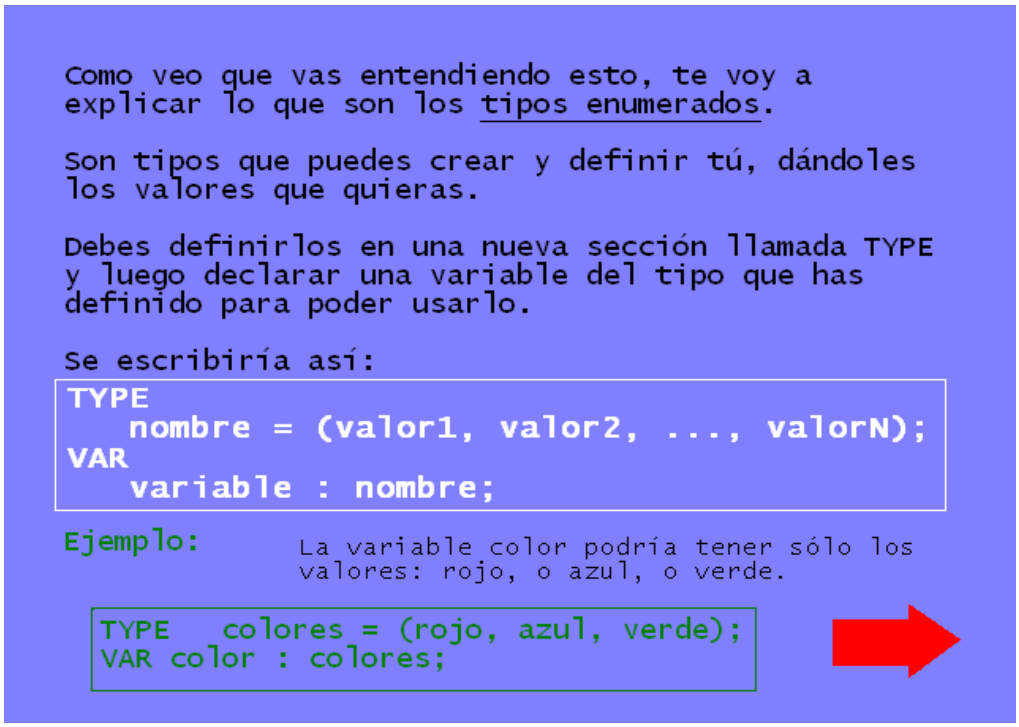


Figura 4.10: Tipos enumerados

Cuando el jugador haya leído la explicación, tiene que pinchar en la flecha roja e irá a esta prueba:

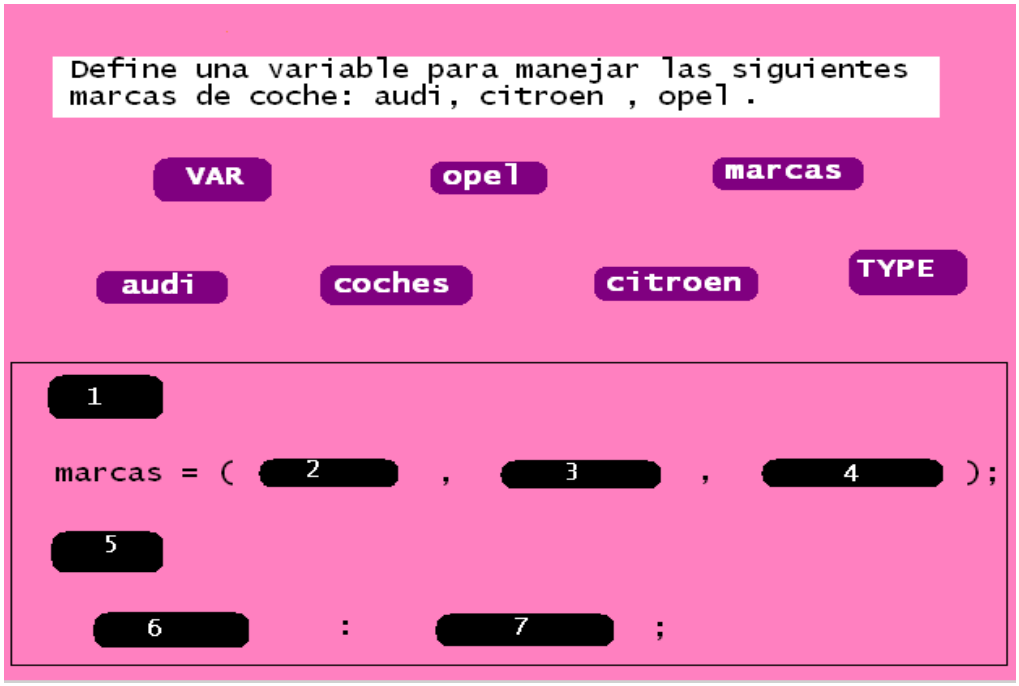


Figura 4.11: Minijuego Tipos enumerados

En la cual al principio aparece la siguiente frase:

"Pincha en los botones morados en el orden que te indican los rectángulos negros"

Para superarla, deberá pinchar en este orden:

1- TYPE
2- Es indiferente el orden de las siguientes tres palabras: audi, opel y citroen
3- VAR
4- coches
5- marcas

Los tipos enumerados servirán al usuario a lo largo del juego, ya que en próximas pruebas deberá usarlos.

4.1.7. Instrucción de selección IF

Este concepto ha sido modificado.

Versión anterior: hacían una pequeña introducción, mediante diálogos, de lo que significaba IF-THEN-ELSE y ponían algún ejemplo. Después, daban paso a un minijuego donde había una balanza (Figura 4.12) y tenías que equilibrarla siguiendo el siguiente código:

"IF (pesoPlatoIzquierdo == pesoPlatoDerecho) has ganado ELSE la balanza esta desequilibrada"

Figura 4.12: Balanza minijuego IF

Pinchando en los números hasta que éstos sumasen doce, se resolvía la prueba.

Errores o problemas: esta idea nos pareció buena, pero escasa.

Versión actual: dejamos este minijuegos intacto. Pero, decidimos ampliar la información que se daba de IF-THEN-ELSE con más minijuegos de ejemplos y pruebas, de la siguiente manera:

Tras superar el juego de la balanza añadimos un diálogo explicatorio tras el cual se presentan las siguientes pruebas:

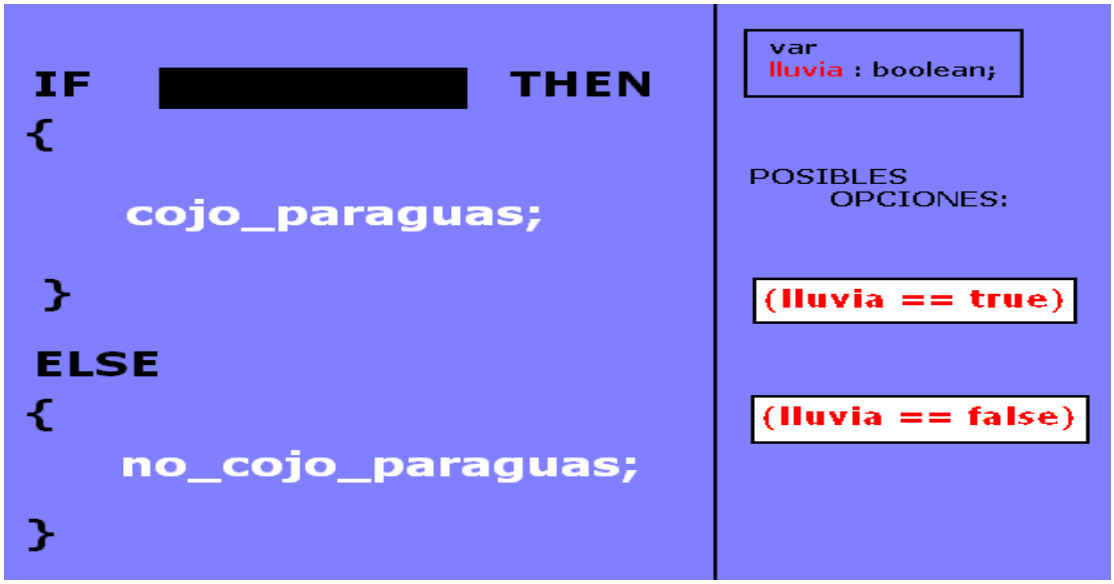
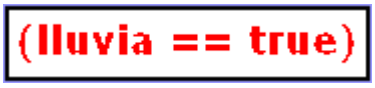


Figura 4.13: Minijuego IF

Al entrar en esta ventana aparece este texto: "Elige una de las dos posibles opciones, la que creas que va en el rectángulo negro."

Para superarla, hay que pinchar en:



Una vez hecho, se da el siguiente consejo al jugador:

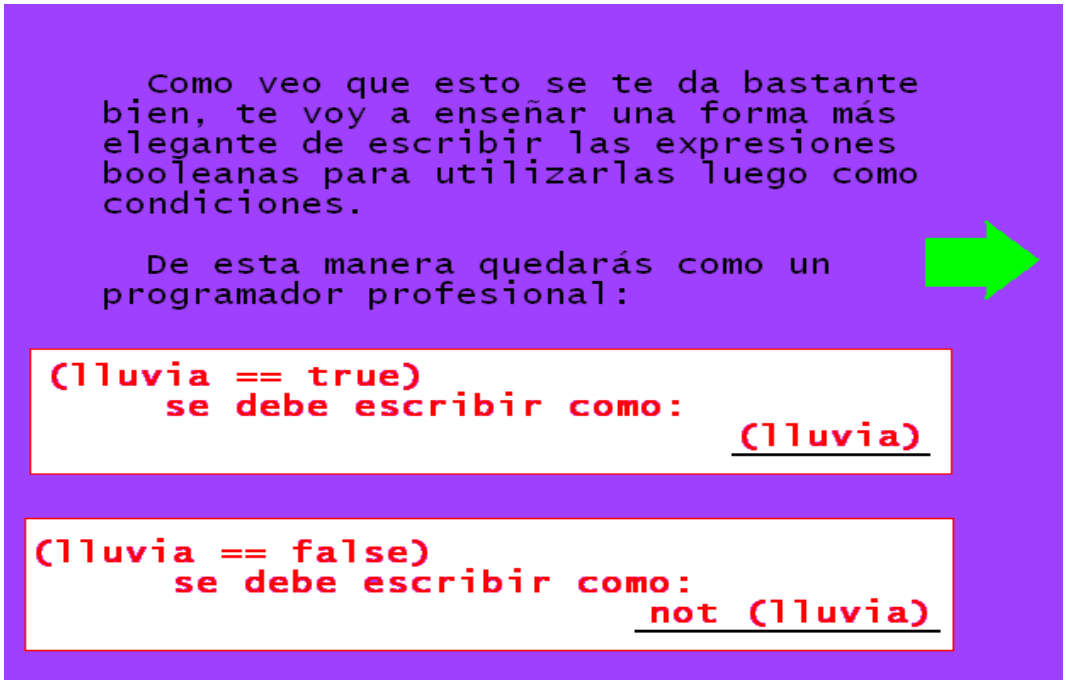


Figura 4.14: Consejo minijuego IF

Y vuelve a repetir la prueba anterior:

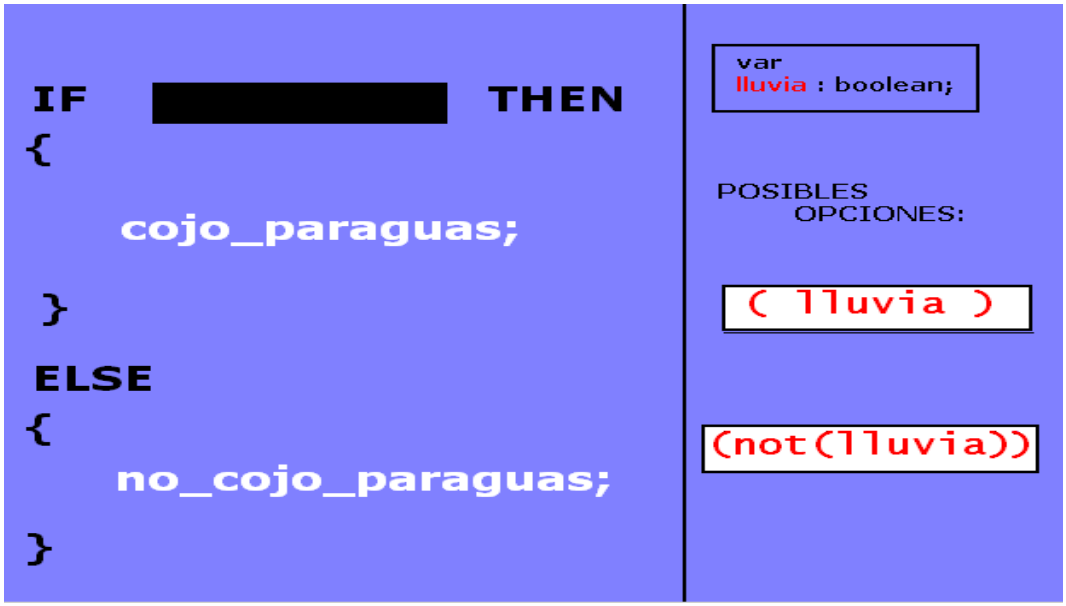


Figura 4.15: Minijuego IF con consejo

Facultad de Informática

Deberá seleccionar:

Después aparecerá este nuevo minijuego:

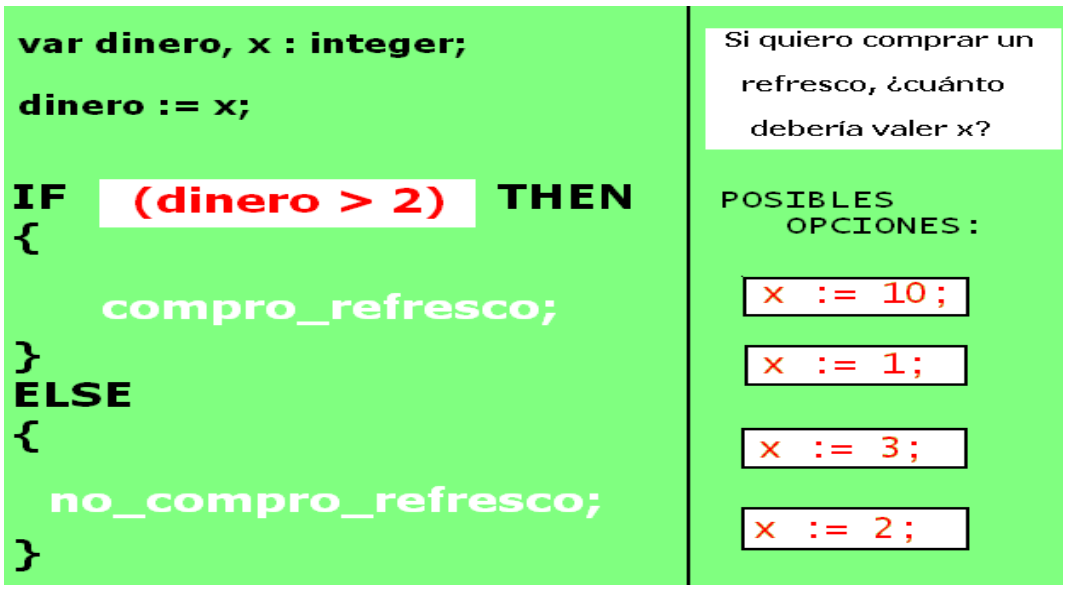


Figura 4.16: Minijuego adicional IF

Para pasarlo, el jugador deberá pulsar o en “x := 10”, o en “x := 3”, y esto nos llevará a la última prueba de esta tanda (Figura 4.17), que tiene el siguiente texto al entrar en ella: "Tienes que ir haciendo click en los elementos en el siguiente orden: " y "1) el que creas que va en el rectángulo, 2) el del circulo de arriba, 3) el del circulo de abajo".

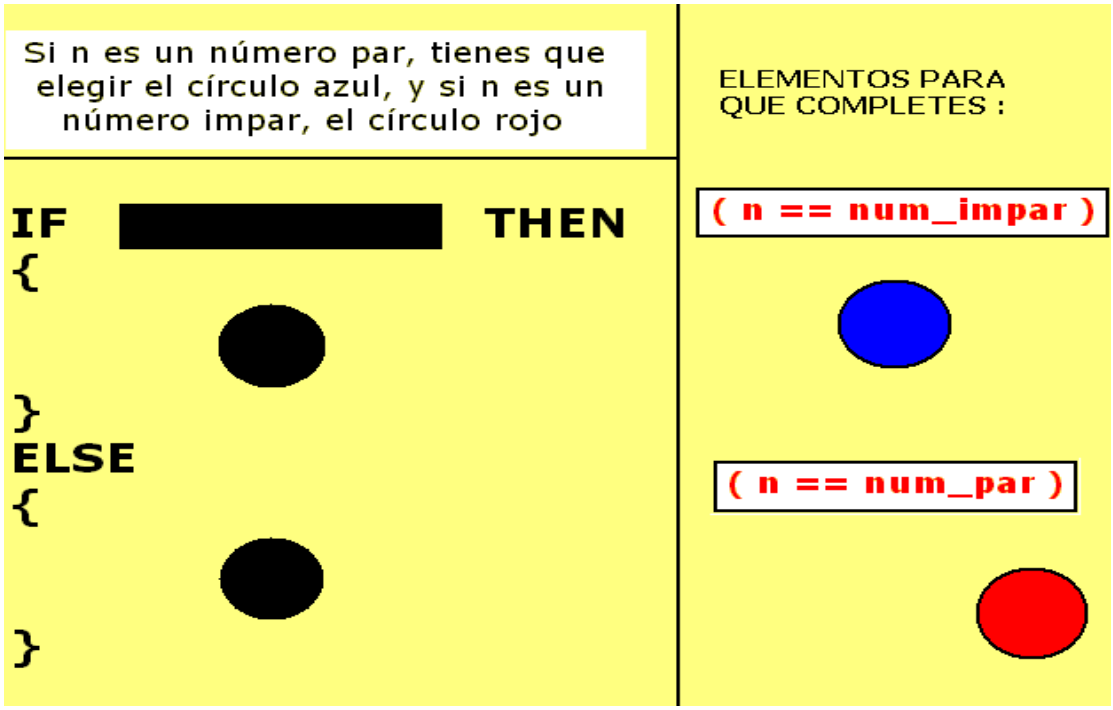
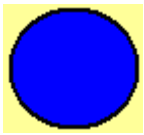
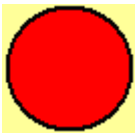


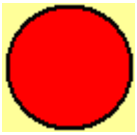
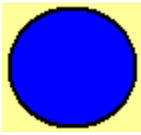
Figura 4.17: Minijuego final IF

Para superarla hay que pinchar en los objetos en alguno de estos dos órdenes:

- Orden 1:

1 → (n == num_par) 2 →  3→ 

- Orden 2:

1 → (n == num_impar) 2→  3→ 

Una vez realizada esta serie de minijuegos, pensamos que este concepto quedará mucho más asentado.

4.1.8. Instrucciones IF anidados, Tipo subrango, CASE

Estos conceptos los introducimos nuevos.

Pensamos que tras explicar el concepto IF, sería conveniente continuar ampliando información y hablar de los IFs anidados y de la sentencia CASE.

Superado el punto anterior (4.1.7.), se muestra un diálogo explicatorio y a continuación el siguiente minijuego, donde usamos el concepto de tipos enumerados:

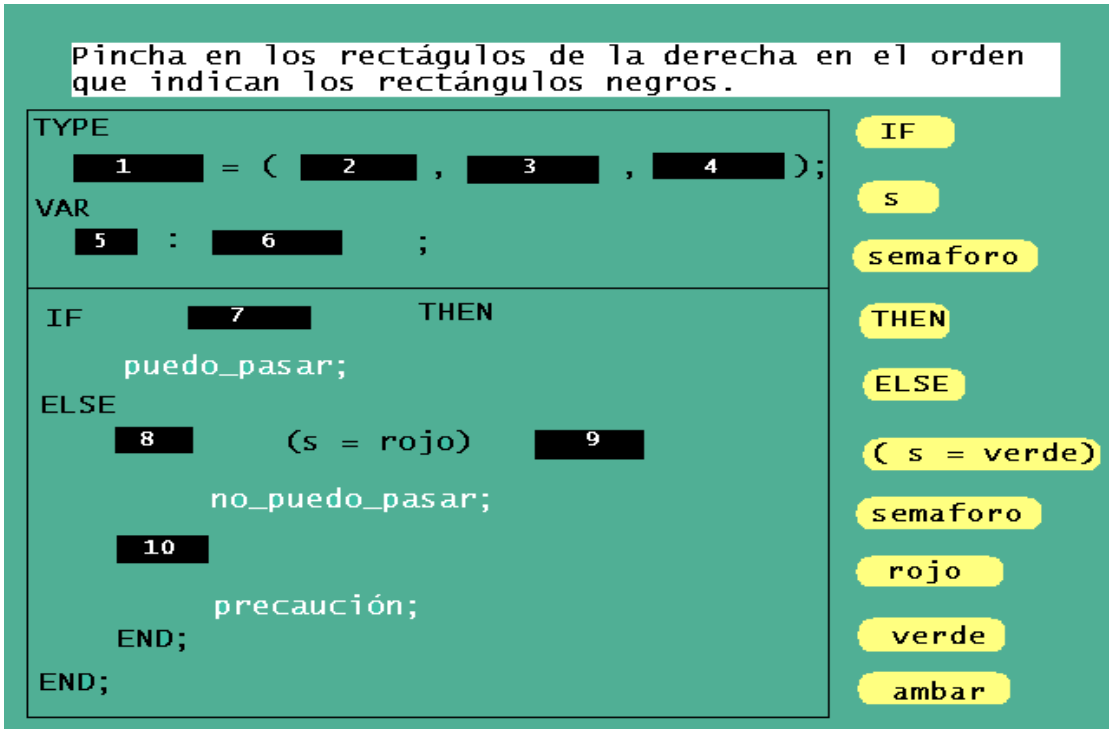


Figura 4.18: Minijuego IF anidados

Esta prueba se resuelve pinchando en los rectángulos amarillos (los cuales se moverán a su correspondiente rectángulo negro) en este orden:

- "semaforo" (cualquiera de los dos)
- Da igual el orden de : "rojo", "verde" y "ambar"

- "s"
- "semaforo" (el que no se haya pulsado en el primero)
- "(s = verde)"
- "IF"
- "THEN"
- "ELSE"

El minijuego anterior, nos lleva a un diálogo que nos dice que varias instrucciones IF-THEN-ELSE anidadas se pueden escribir como un CASE así:

"CASE variable OF
valor1cte: sentencias;
valor2cte: sentencias;
…
else sentecias;"

y luego introducimos el concepto de tipo subrango con un diálogo (mostramos parte del código fuente del diálogo):

```
carmen: Lo de valor1cte significa que puede ser una constante (de los tipos que ya viste)
o un subrango
carmen: Un SUBRANGO se escribe: valorInicio..valorFin
carmen: por ejemplo: para referirme a los números del 1 al 5, escribiría: 1..5
```

Facultad de Informática

Después de esto, en las figuras 4.19 y 4.20 explicamos al usuario cómo poner el ejemplo de los tres grupos de números, con IFs anidados y con CASE, tras lo cual se pasará al minijuego mostrado en la figura 4.21.

Figura 4.19: Explicación IF anidados

Figura 4.20: Explicación CASE

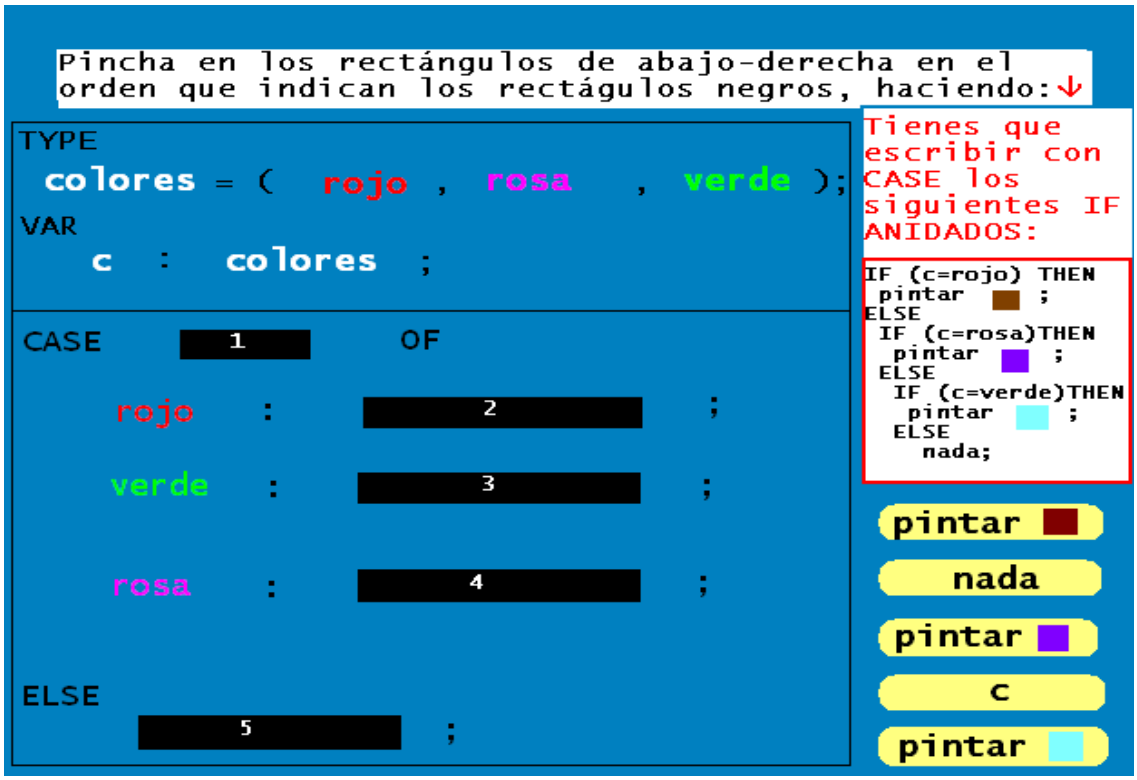
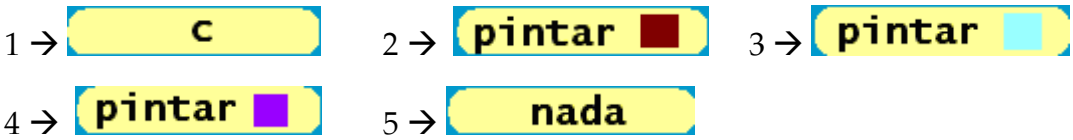


Figura 4.21: Minijuego CASE

Se resuelve pinchando en los rectángulos amarillos en el siguiente orden:



4.1.9. While

Este concepto ha sido modificado.

Versión anterior: se hacía que el usuario intentara atravesar un laberinto pulsando en la entrada del mismo, un botón. Este botón, iniciaba la cuenta atrás e iba haciendo que una verja que hay en la salida se fuese cerrando.

La idea es que al pulsar el botón, el tiempo se inicializa a quince y se va restando uno, entonces mientras el tiempo sea mayor que cero, el personaje podrá cruzar la verja de salida del laberinto. En caso de que no lo consiguiera, habría que volver a empezar el proceso.

Errores o problemas: consideramos que la explicación anterior transmite claramente el concepto de bucle while. Pero, decidimos que había que retocar la forma de exponerlo en el juego, ya que hacía lo siguiente: cuando se conseguía atravesar el laberinto, un personaje del juego (el bedel) contaba a Lucas, que lo que acababa de hacer se podía representar en código e iba poniendo línea a línea el trozo de programa, mediante diálogos. Con lo cual, cuando aparecía una línea nueva, la anterior ya se había borrado y era muy difícil obtener una visión general de lo que había ocurrido.

Versión actual: debido a los problemas comentados anteriormente, decidimos poner una serie de tres pantallas, que contuviesen el código completo y que comprobasen si se ha entendido el concepto.

En la primera (Figura 4.22), aparecerá todo el código correspondiente al trozo de programa mencionado en párrafos anteriores. Así, el usuario podrá leerlo todo el tiempo que necesite. Cuando decida continuar, pinchará la flecha negra que le llevará a la siguiente pantalla (Figura 4.23).

El objetivo de este juego es comprobar que el usuario ha entendido cómo funciona un bucle while. Damos la opción de que pueda volver al código del laberinto para consultarlo.

Cuando el usuario pulse en el "3", que es la respuesta correcta, pasará automáticamente a la última pantalla (Figura 4.24) donde simplemente damos una pista de como ir a la Facultad de Matemáticas.

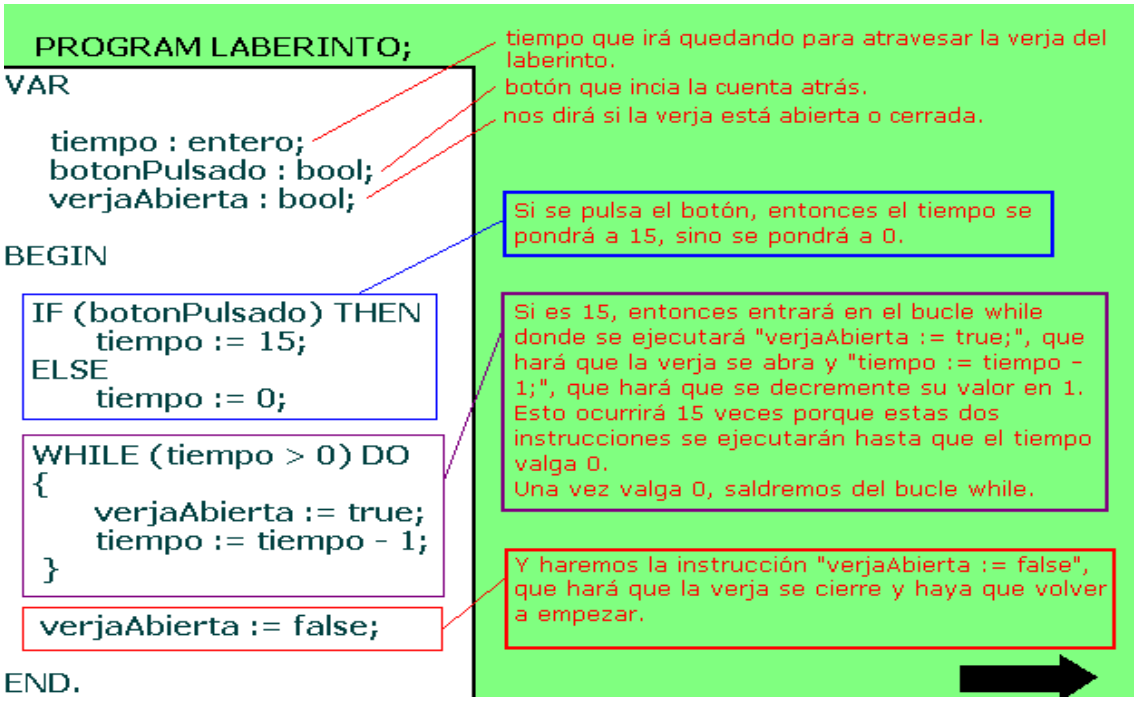


Figura 4.22: Código WHILE Laberinto

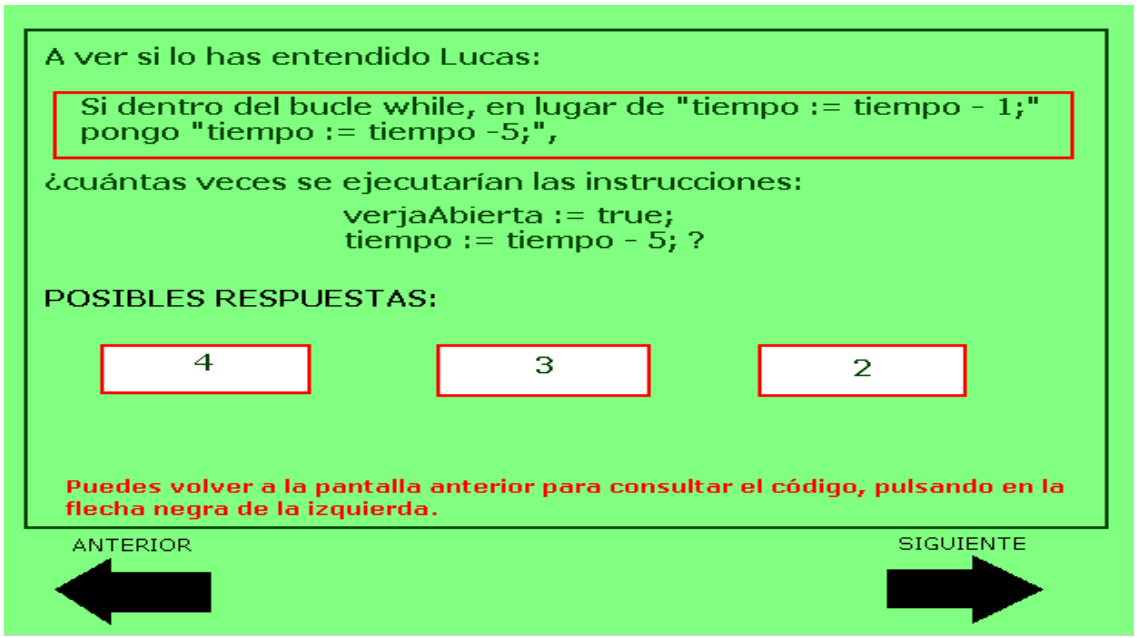


Figura 4.23: Minijuego While

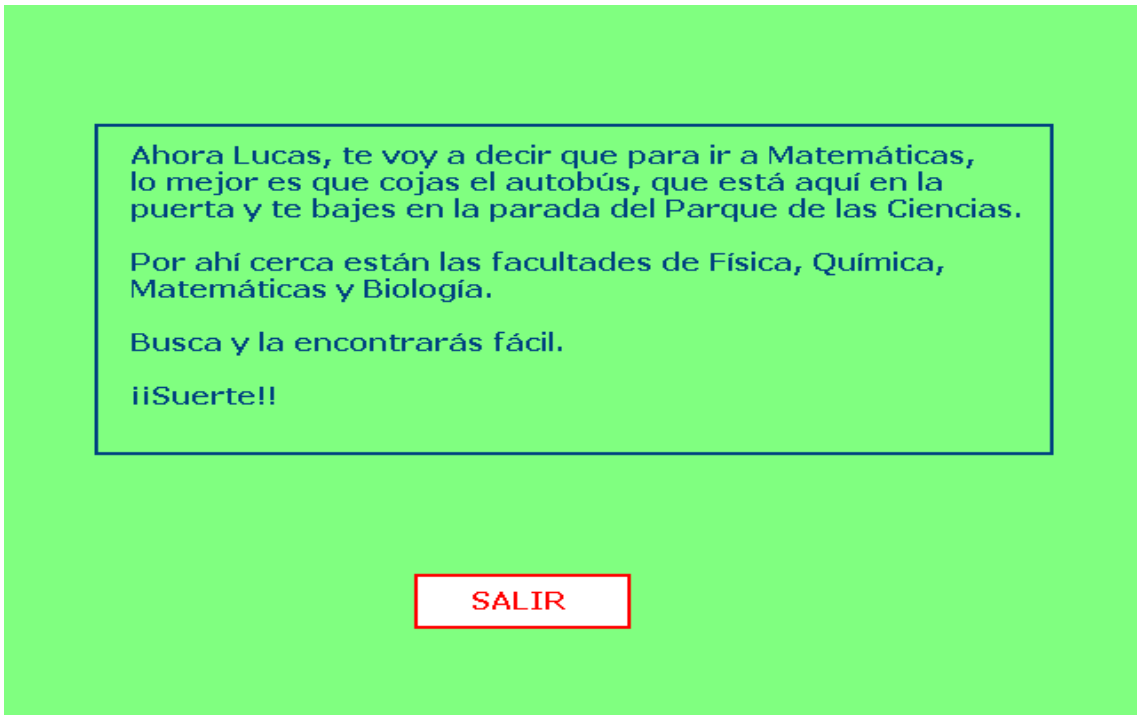


Figura 4.24: Camino a Matemáticas

4.1.10. Instrucción FOR

Este concepto ha sido modificado.

Versión anterior: se hacía una prueba al usuario, en la cual debía hacer lo que le decía el bucle for, que aparece escrito en la Figura 4.25.

Errores o problemas: como se puede observar, en el cuerpo del bucle for pusieron "i:=i+1", lo cual puede ser un error dependiendo del lenguaje de programación del que hablemos.

Versión actual: simplemente quitamos esa instrucción del cartel, ya que el resto de la prueba nos parecía que transmitía de forma correcta el concepto de bucle for.

Figura 4.25: Juego antiguo FOR

4.1.11. Punteros

Este concepto lo introducimos nuevo.

El concepto de puntero es una parte de la programación que, por lo general, cuesta entender. Por esta razón decidimos incluirlo en nuestro proyecto. Así, al dar un pequeño avance de lo que es un puntero, probablemente cuando las personas que hayan jugado tengan que estudiarlo les será más fácil comprenderlo. En este caso se pretende dar una idea lo más intuitiva posible, para que no se vea esto como algo "extraterrestre".

Primero añadimos un diálogo como introducción a dos pantallas nuevas. En la primera (Figura 4.26) explicamos el concepto de puntero, usando la idea que ya

teníamos de variable. En la segunda (Figura 4.27) hacemos un minijuego para ver si se ha entendido el concepto. Para superarlo, el usuario deberá pinchar en los rectángulos blancos y azules en el siguiente orden:

- cualquiera que ponga "int"
- cualquiera que ponga "real"
- cualquiera que ponga "char"
- cualquiera que ponga "bool"
- el que quede de los dos "int"

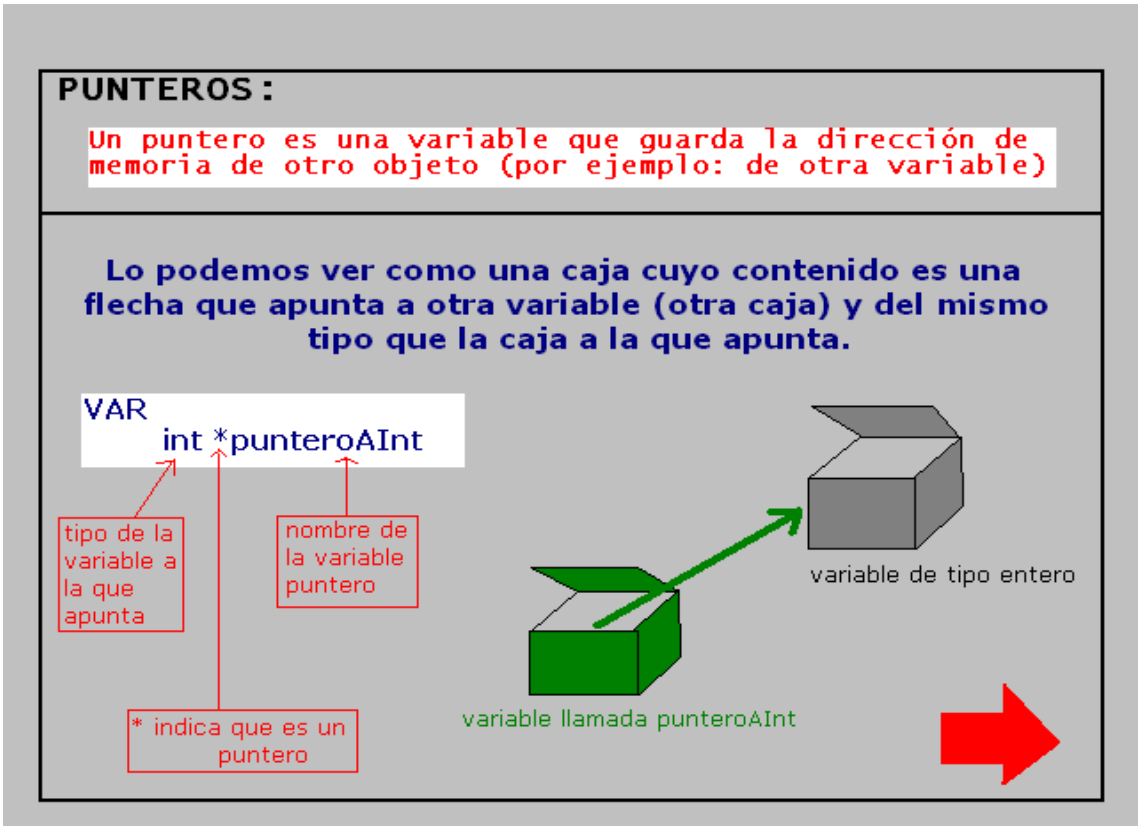


Figura 4.26: Explicación Punteros

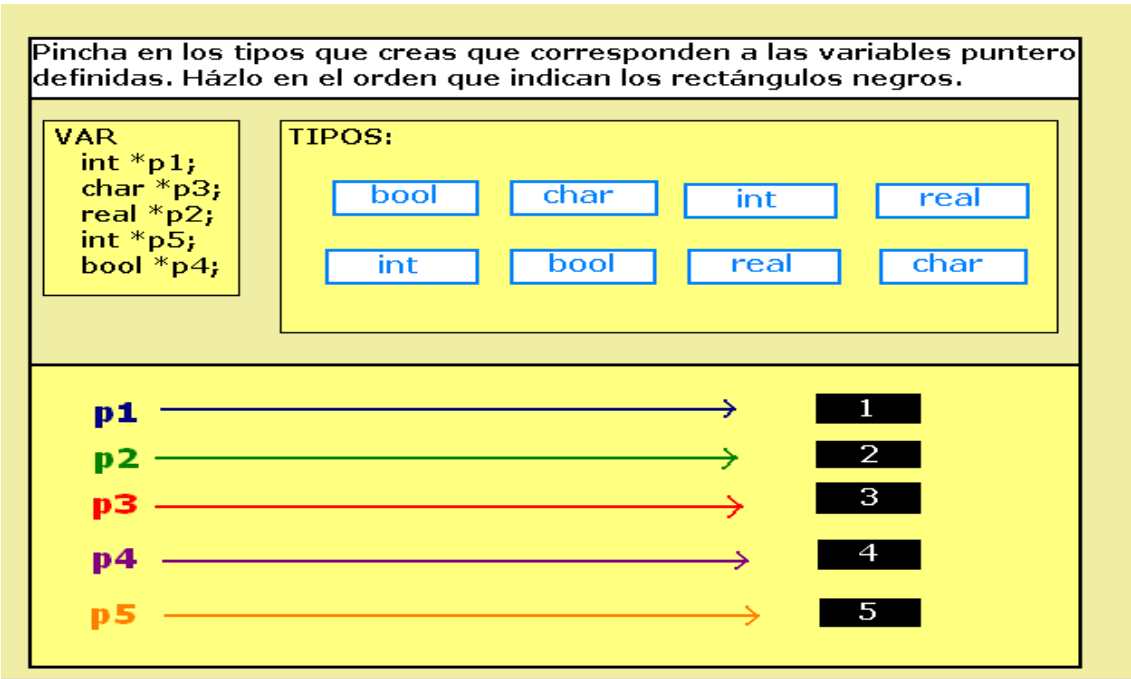


Figura 4.27: Minijuego Punteros

4.1.12. Listas, Pilas y Colas

Este concepto lo introducimos nuevo.

Cuando decidimos incluir punteros en el proyecto, necesariamente hubo que incluir también una pequeña parte de memoria dinámica. Pensamos en la implementación dinámica de los tipos de datos listas, pilas y colas, ya que se pueden explicar de una manera sencilla y con ejemplos gráficos, que expresan de forma directa la idea intuitiva que se esconde tras estos conceptos.

También vimos que introducir esta parte, sería de gran utilidad para una comprensión más profunda del concepto de puntero, puesto que en esta implementación de las listas, las pilas y las colas se usan punteros.

Facultad de Informática

Pasamos a mostrar las tres pantallas creadas, una para cada tema: Listas (Figura 4.28), pilas (Figura 4.29) y colas (Figura 4.30)

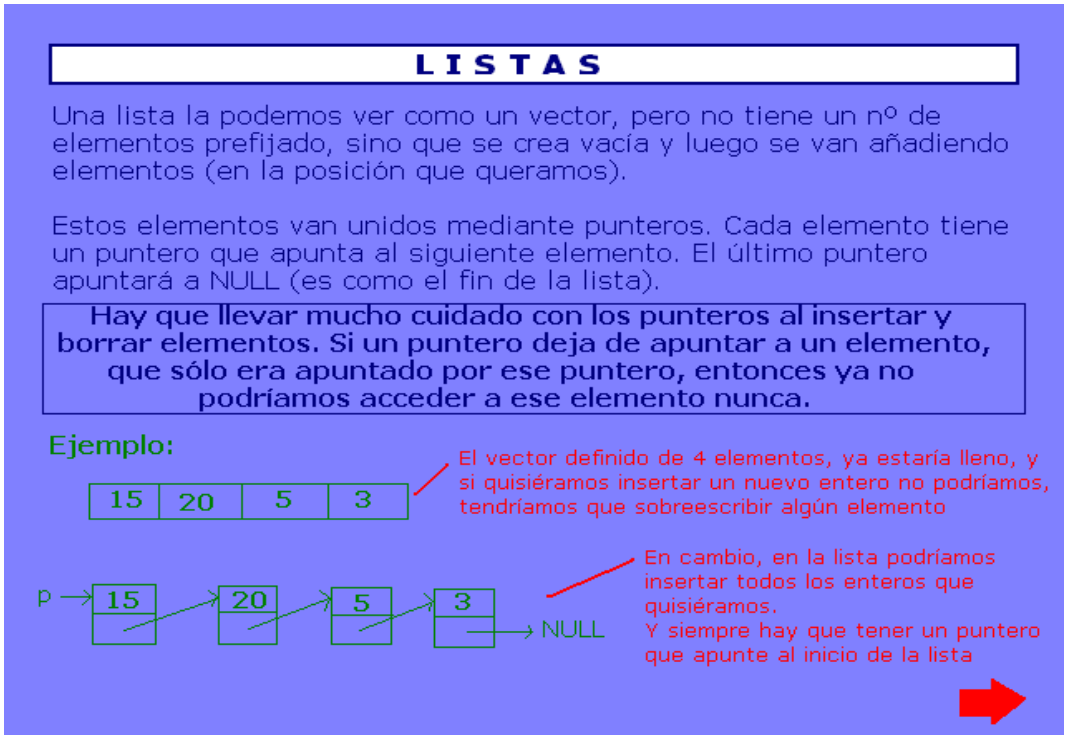


Figura 4.28: Listas

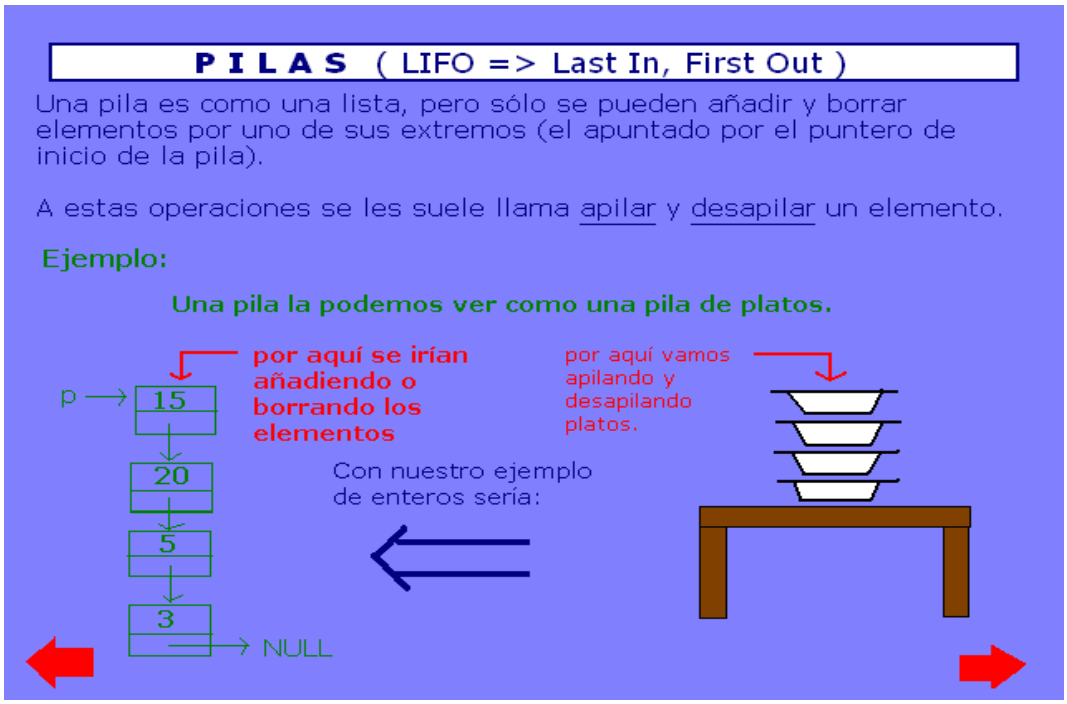


Figura 4.29: Pilas

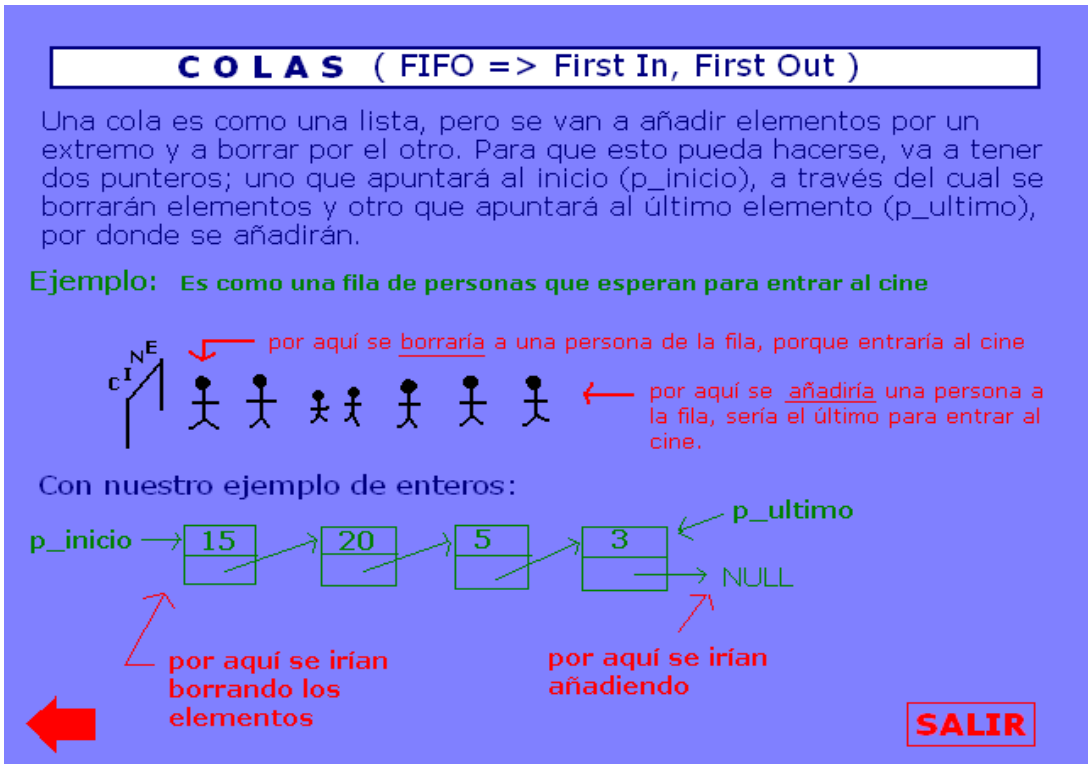


Figura 4.30: Colas

4.1.13.Funciones y procedimientos

Este concepto lo introducimos nuevo.

Toda buena guía de programación contiene un apartado en el que se explica el concepto de la programación modular o subprogramas. Llamémoslo como queramos pero todos sabemos que nos referimos a los procedimientos y funciones. Por lo tanto, como nuestro objetivo era instruir a los jugadores de nuestro juego no podíamos pasar por alto estos conceptos. Se trata de una ampliación realizada en la segunda fase de realización de esta aventura gráfica. Éstos son los últimos términos que se explican y se añaden a *"el gran libro de la programación"*, que se va facilitando al jugador según va adquiriendo nuevas nociones sobre programación.

Al ser el último concepto, es lógico que aparezca casi al final del juego, concretamente es Eva la que como agradecimiento a que Lucas le ha arreglado el ascensor para que pudiera bajar, ya que se encuentra lisiada, le empieza a comentar lo que es la programación modular y le da acceso a las habitaciones con las definiciones y juegos que completan la enseñanza de este término.

Metiéndonos en las entrañas del juego, vamos a explicar como conseguimos que funcione esta parte:

→ Como es una ampliación sobre la versión realizada en la primera temporada, optamos por no modificar los valores de los estados de "EVA" para introducir esta parte. Por tanto, creamos una nueva variable booleana global, "*Procedimientosfin*" que nos indicará si Eva le ha explicado la programación modular o no.

→ Si Eva ha bajado por el ascensor, salta el dialogo que inicia las explicaciones y al terminar esa conversación nos lleva a la pantalla de definición de procedimientos.

→ Al finalizar el juego de las funciones, el juego continúa por el camino que explicamos al final del punto anterior.

Este apartado ocupa cuatro habitaciones nuevas en el juego:

→ Definición de Procedimientos

→ Juego sobre Procedimientos

→ Definición de Funciones

→ Juego sobre Funciones

A continuación, mostramos las cuatro habitaciones en las que se incluyen las anotaciones precisas para su completa asimilación.

DEFINICIÓN SOBRE PROCEDIMIENTOS:

Desde un primer momento, pensamos que la mejor forma de que el jugador captara los conceptos de "procedimiento" y "función" sería mostrando una nueva ventana (Figura 4.31) en la que incluyéramos los puntos importantes de estas definiciones, y que el usuario pudiera leer tranquilamente, sin las prisas que comprendimos que surgían cuando el jugador leía los diálogos, ya que estos desaparecen tras un tiempo. En cambio, mediante la ventana, es el usuario el que cuando ha comprendido bien el concepto podría avanzar.

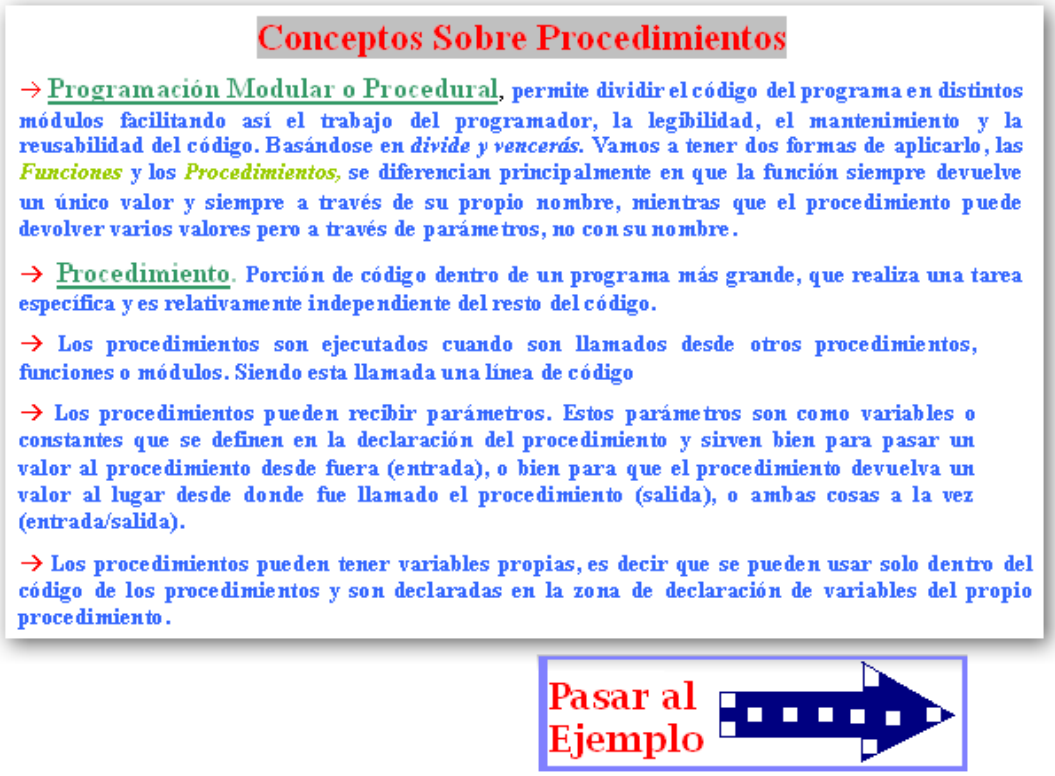


Figura 4.31: Conceptos sobre procedimientos

Facultad de Informática

Inicialmente en esta ventana no se puede ver un botón cuya utilidad es llevar al usuario a la definición de función, ya que éste solo sale si ya se ha visitado esa habitación. Esto, está pensado porque como se puede ver en la imagen, el siguiente paso es acceder al ejemplo, es decir, al juego del procedimiento. De este modo no permitimos al jugador ir desde la definición del procedimiento a la de la función sin realizarse el juego del procedimiento. Por lo tanto, si desde la definición de la función volvemos a la del procedimiento, por ejemplo para comparar los conceptos, podremos observar el siguiente botón en esta habitación.

JUEGO PROCEDIMIENTOS:

Como en el resto del juego, comprendimos que una definición sin un ejemplo práctico en un juego como el que intentábamos desarrollar no tenía sentido. Por ello, en este punto hemos optado por simular un caso real de código como ejemplo práctico. Concretamente, diseñamos un juego (Figura 4.32) en el que el jugador tuviera que ir poniendo las partes que faltan en el orden adecuado para construir un programa que contenía un procedimiento, que realizaba la suma de dos números. Los elementos que hay que añadir son objetos (sprites). Por lo tanto, el objeto que teníamos que ir poniendo en cada posición siguiendo el orden es:

→ **1. ProgPrincipal**. Es el nombre del programa principal.

→ **2. ProcSuma**. Es el nombre del procedimiento, lo podemos intuir ya que es el nombre utilizado en el programa principal para invocarlo.

→ **3. S1**. Representa al primer parámetro del procedimiento y viene a ser el primer sumando de la suma que se realiza dentro del procedimiento.

→ **4. S2**. Representa al segundo parámetro del procedimiento y viene a ser el segundo sumando de la suma que se realiza dentro del procedimiento.

→ **5. Resultado** . Es la variable local del procedimiento, se usa para almacenar el resultado de la suma y luego asignárselo al parámetro de salida.

Si hemos logrado poner cada cosa en su sitio, automáticamente pasaríamos a la habitación en la que se muestra la definición de las funciones.

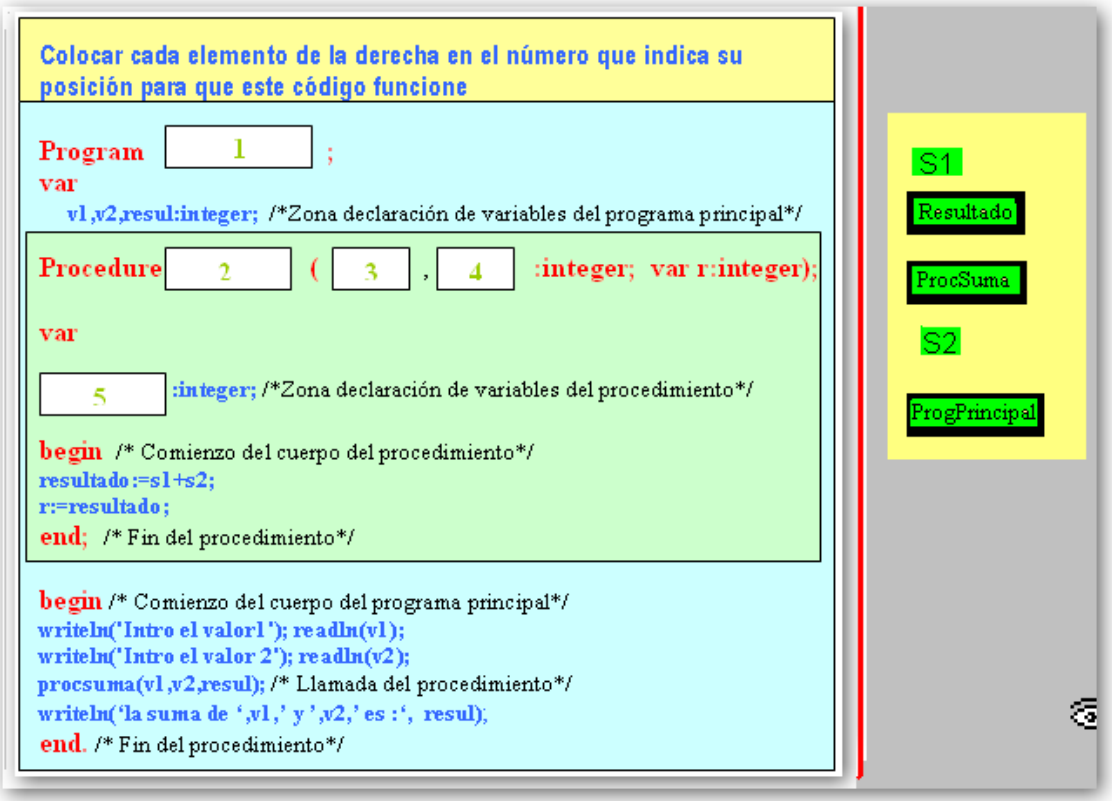


Figura 4.32: Minijuego procedimientos

DEFINICIÓN FUNCIONES:

No podíamos explicar los procedimientos y dejar de lado las funciones. En este apartado, se define el concepto de función (Figura 4.33) y a continuación

pasaríamos a realizar un ejemplo (Figura 4.34). Como se puede ver, desde aquí también podemos retroceder a la definición de procedimiento.

Figura 4.33: Conceptos sobre funciones

JUEGO FUNCIONES:

Se trata de realizar la misma operación que se hizo con los procedimientos solo que esta vez con funciones, es decir, tenemos que ir poniendo en orden los objetos de la derecha sobre el código facilitado.

En esta ocasión el objeto que teníamos que ir poniendo en cada posición siguiendo el orden es:

→ **1. ProgPrincipal**. Es el nombre del programa principal.

→ **2. FunSuma**. Es el nombre de la función, lo podemos intuir ya que es el nombre utilizado en el programa principal para invocarla.

→ **3. S1**. Representa al primer parámetro de la función y viene a ser el primer sumando de la suma que se realiza dentro de la función.

→ **4. S2**. Representa al segundo parámetro de la función y viene a ser el segundo sumando de la suma que se realiza dentro de la función.

→ **5. Resultado** . Es la variable local de la función, se usa para almacenar el resultado de la suma y luego asignárselo a la propia función como valor que va a devolver.

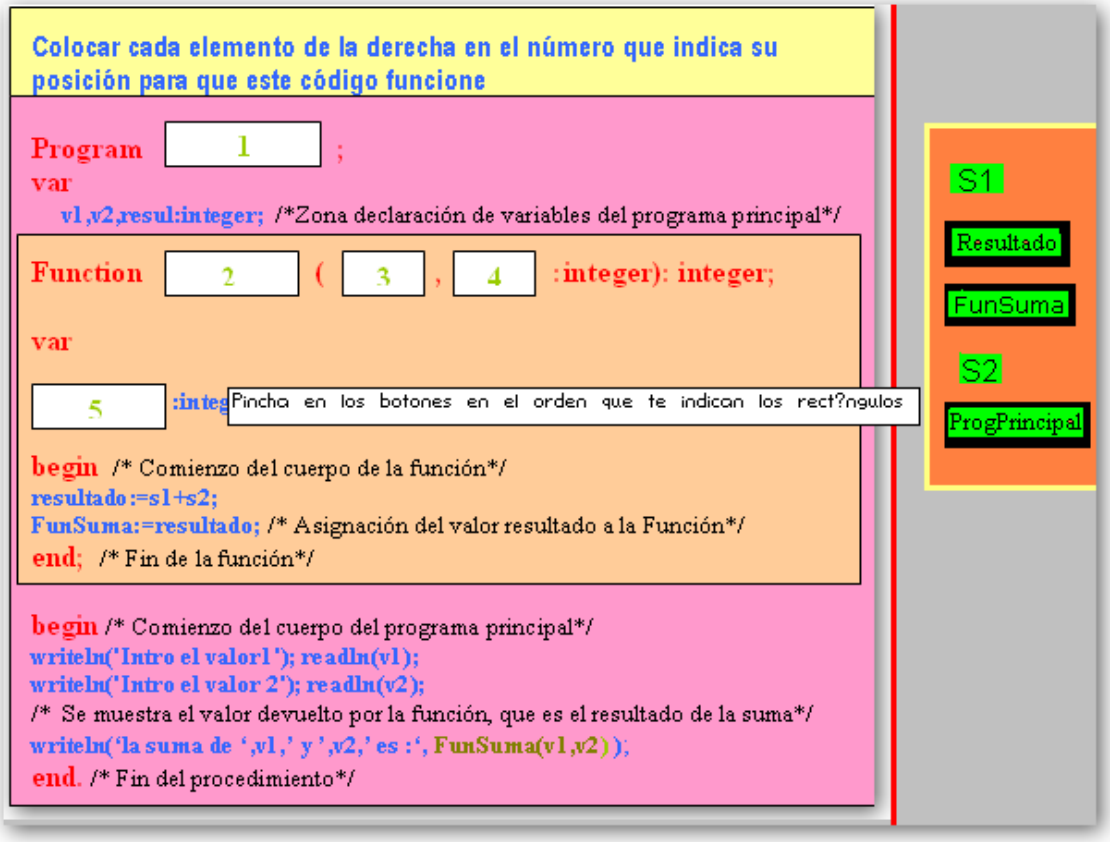


Figura 4.34: Minijuego funciones

Una vez superado este juego la variable global "procediemientosfin" toma el valor true, lo que significa que la programación modular ya ha sido explicada y se volvería al juego allí por donde lo habíamos dejado.

4.1.14.Repaso final

Este aspecto ha sido modificado.

Versión anterior: hicieron lo mismo que con el test de las asignaciones, contado en el punto "4.- Concepto: ASIGNACIONES", pero usando un test que tocaba todos los conceptos explicados en el juego.

Errores o problemas: de nuevo, las preguntas pasaban muy rápido, por estar hechas con diálogos y se utilizaba la función random(). Con lo cual, a veces, se repetía la misma cuestión.

Versión actual: la solución a estos problemas es la misma que antes. Ponemos dos cuestiones por página con el programa Paint y cuando el usuario pinche en la solución correcta de las dos preguntas, pasará automáticamente a la siguiente pantalla.

Las cuestiones que usamos son, por un lado, modificaciones de las del año pasado, por otro, copia directa y el resto nuevas. Las nuevas, se han incluido porque era necesario, al haber incluido nuevos conceptos en esta nueva versión.

Para superar la primera pantalla (Figura 4.35), deberá pulsar en la cuestión de la izquierda el "16" y en la de la derecha, "nunca saldría del bucle → bucle infinito".

Facultad de Informática

Para superar la segunda (Figura 4.36) deberá pulsar en la cuestión de la izquierda, el "2" y en la de la derecha, "0".

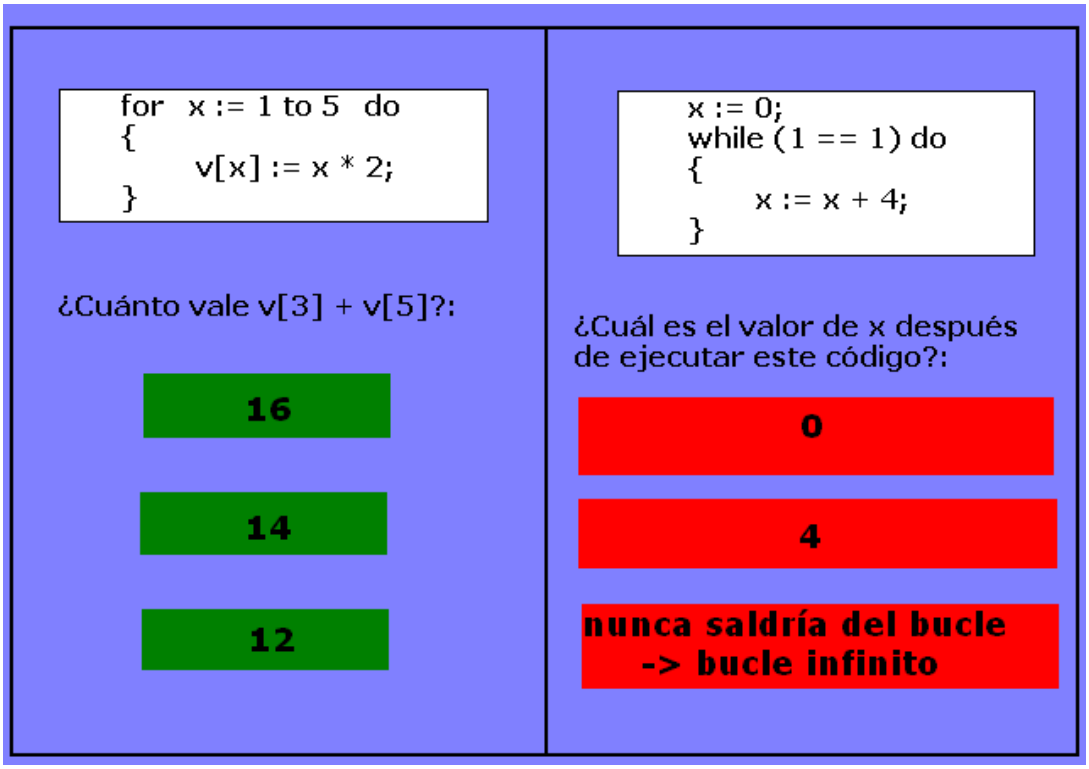


Figura 4.35: Cuestionario final, parte 1

Para pasar la tercera (Figura 4.37) deberá pulsar en la cuestión de la izquierda, el "hay un error de tipos" y en la de la derecha, "false". Para superar la cuarta (Figura 4.38) deberá pulsar en la cuestión de la izquierda, el "10" y en la de la derecha, "60". Finalmente, para pasar la quinta (Figura 4.39) deberá pulsar en la cuestión de la izquierda, el "3" y en la de la derecha, "game".

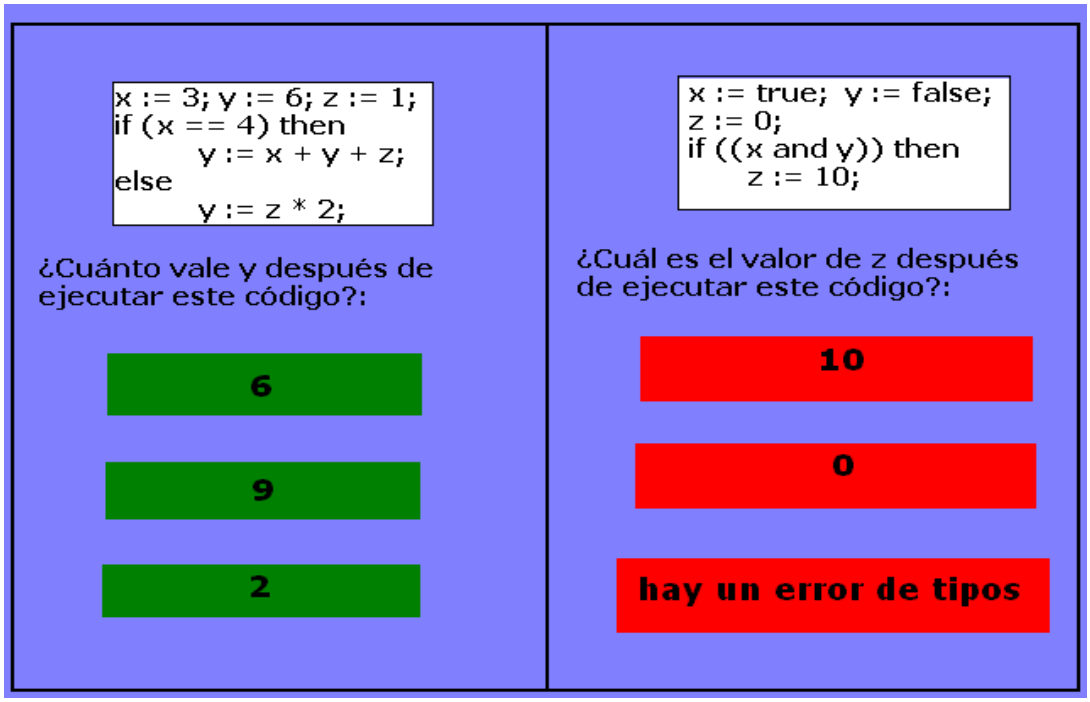


Figura 4.36: Cuestionario final, parte 2

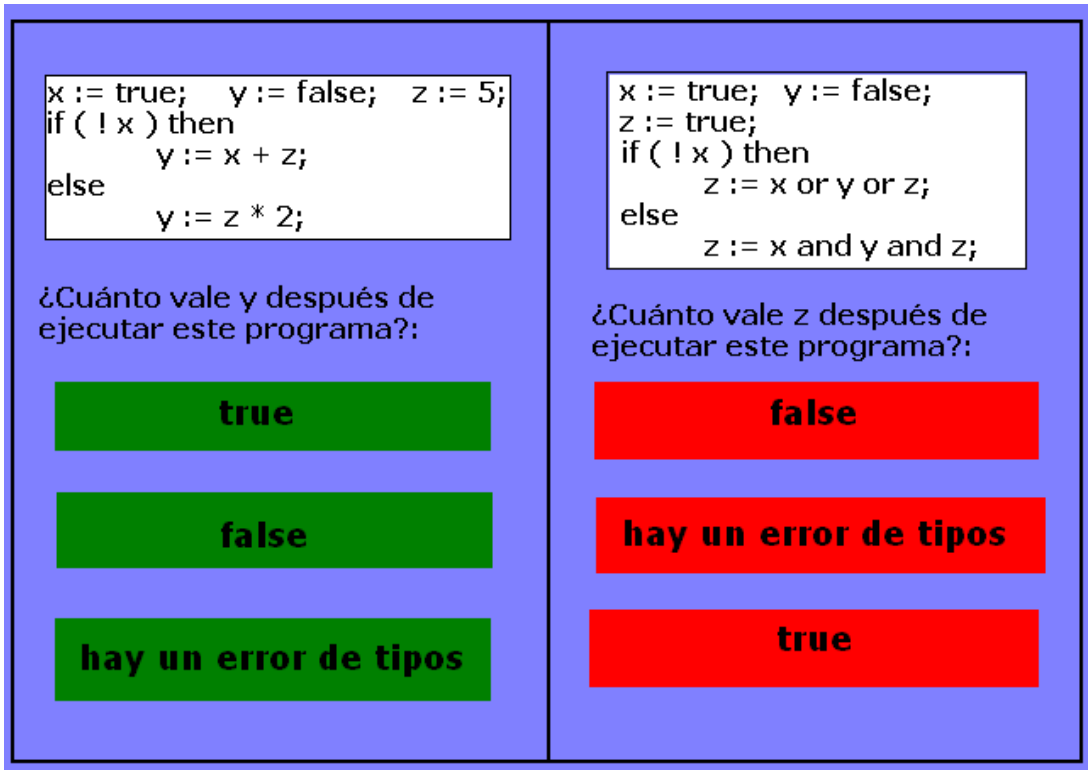


Figura 4.37: Cuestionario final, parte 3

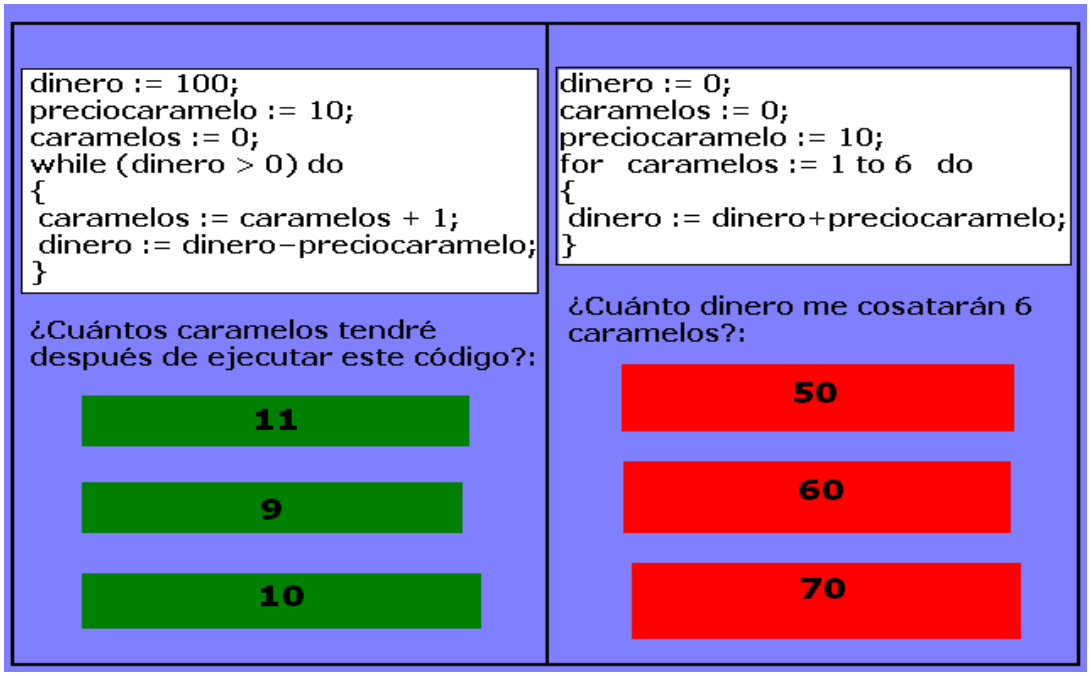


Figura 4.38: Cuestionario final, parte 4

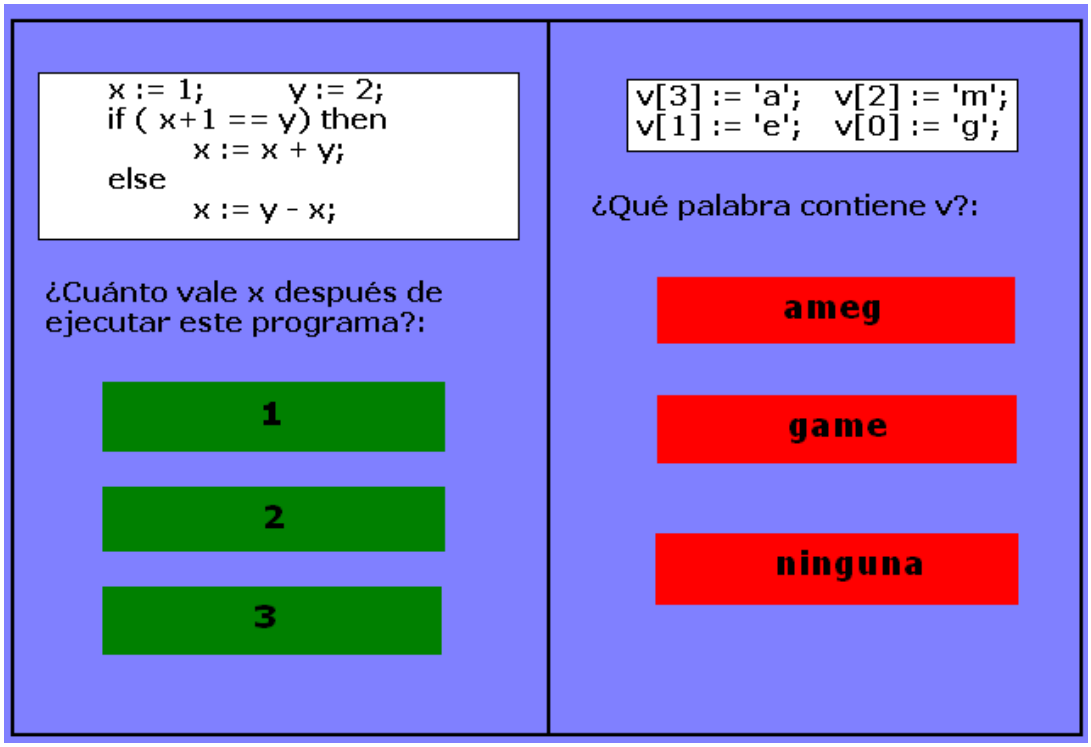


Figura 4.39: Cuestionario final, parte 5

4.2. Traducción

Como ya hemos mencionado anteriormente, tratándose de un juego educativo dirigido especialmente a estudiantes universitarios, uno de los principales objetivos que nos propusimos fue precisamente el de suministrar a dichos alumnos la posibilidad de jugar en inglés.

Además, todos los programas internacionales proporcionados por las Universidades hacen que una parte considerable de los posibles usuarios sean extranjeros.

El AGS no proporcionaba ninguna utilidad para llevar a cabo esta traducción total de la aventura gráfica, por lo que recurrimos a un variable que indica el estado (ingles=false, ingles=true) y en función de la cual se lanza el juego en un idioma o en otro. Dicha traducción debía llevarse a cabo en:

- **Diálogos:** Todos los diálogos definidos como tal en el AGS fueron duplicados siguiendo el mismo esquema de opciones que los originales, e integrados en el código (Figura 4.40).

- **Minijuegos:** Como hemos explicado anteriormente, todos los escenarios son definidos en el AGS como 'Rooms'. Con las pantallas de los diferentes juegos (IF, Pilas, Funciones…) sucede lo mismo. Muchas de ellas tienen explicaciones en forma de texto, por lo que hubo que traducir también todas estas 'Rooms' (Figura 4.41).

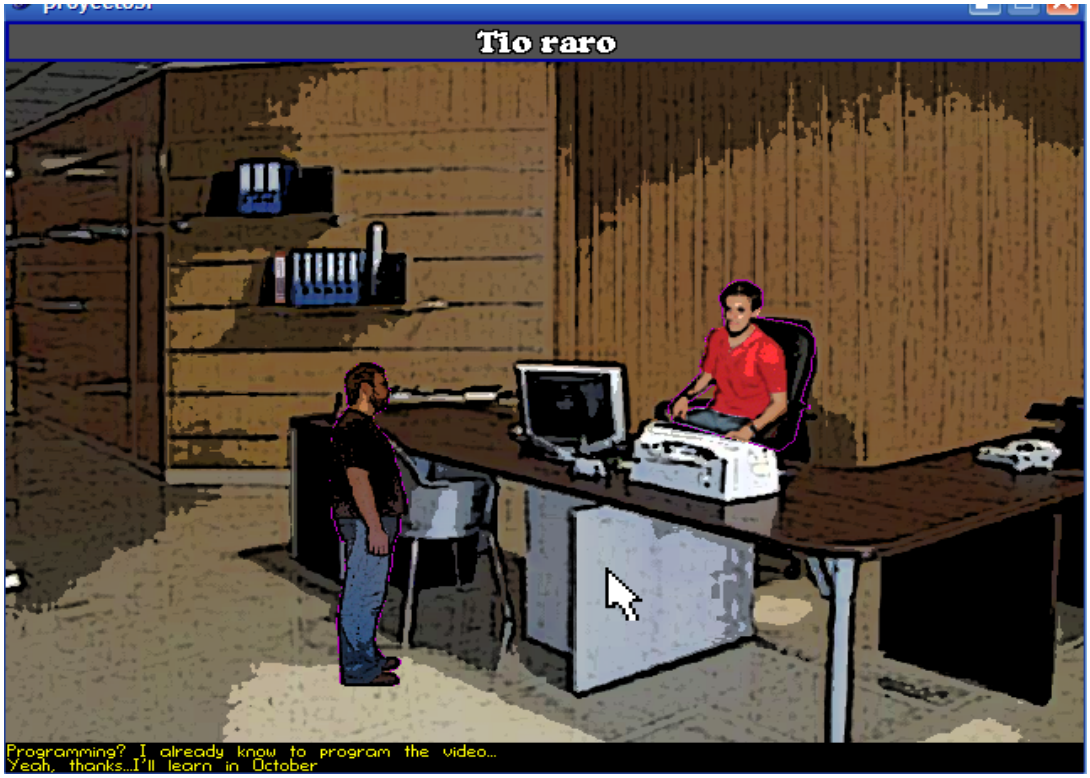


Figura 4.40: Ejemplo traducción diálogos

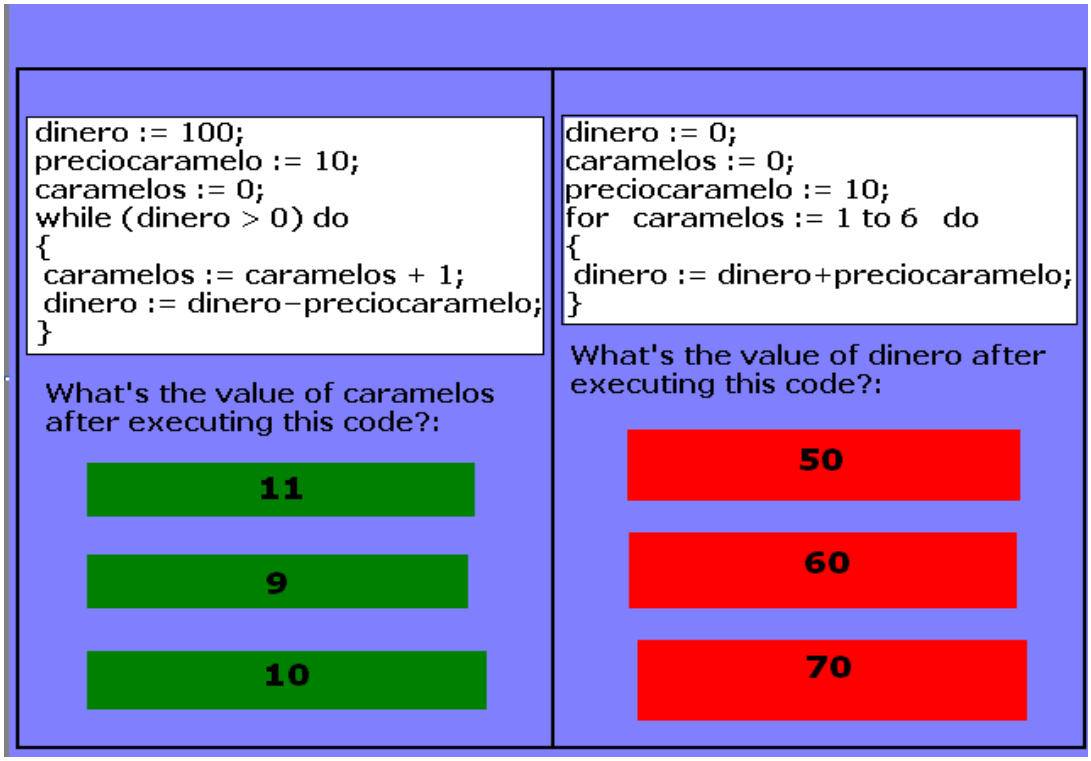


Figura 4.41: Ejemplo traducción minijuego

- **Conversaciones integradas en el código:** Son diálogos entre personajes como los ya mencionados, pero estas están integrados en el código normal y no suponen una interacción con el usuario. De la misma forma que los diálogos normales, fueron también duplicados y traducidos al inglés (Figura 4.42).

Figura 4.42: Ejemplo traducción conversación

- **Gran Libro de la Programación:** Es una 'Room' especial en la que se van agregando objetos a medida que avanza el juego. Se duplicó y tradujo completamente (Figura 4.43).

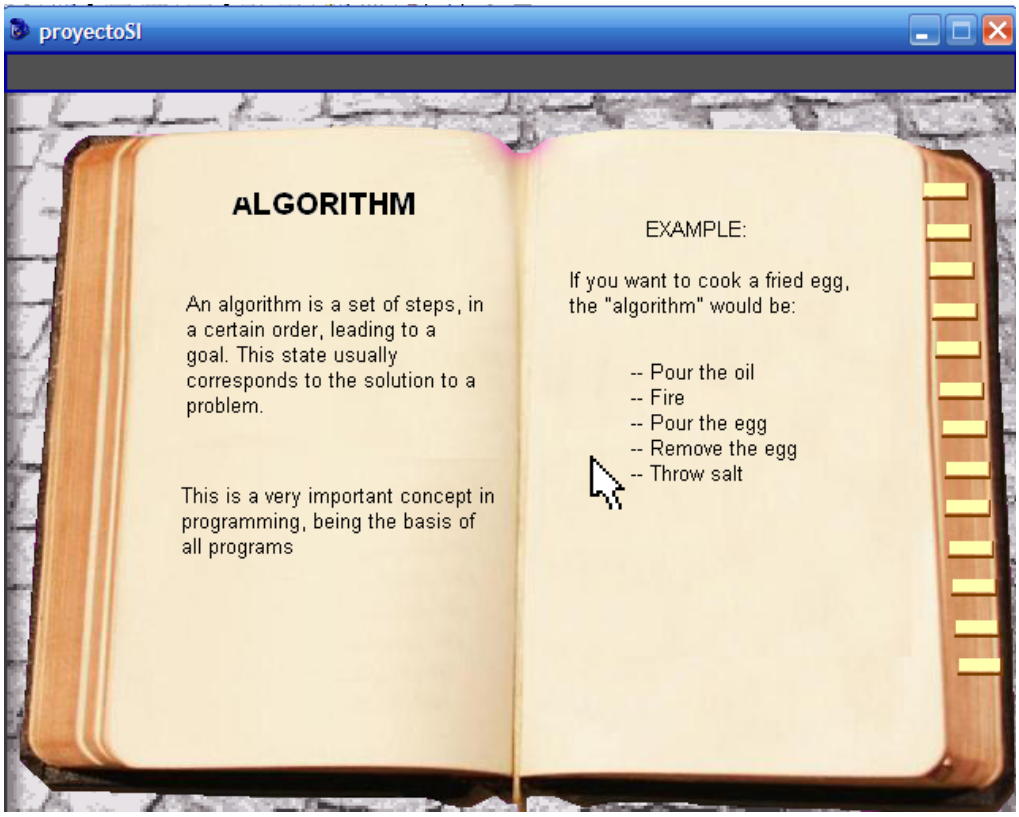


Figura 4.43: Ejemplo traducción Gran libro de la Programación

- **Hotspots:** Una de las formas de saber hacia dónde dirige un camino, o si es posible realizar una acción sobre algún objeto, es mediante descripciones que aparecen en la parte superior de la pantalla. Estas descripciones aparecen en el código en forma de hotspots, y fue necesario traducir también todos sus nombres. Sin embargo, como el AGS no permite condicionar dichos nombres en función del idioma, tuvimos que hacer que apareciesen siempre en ambos lenguajes (Figura 4.44).

Figura 4.44: Ejemplo traducción hotspots

- **Ayuda:** También todas las pistas proporcionadas en la sección Ayuda debían ser traducidas (Figura 4.45).

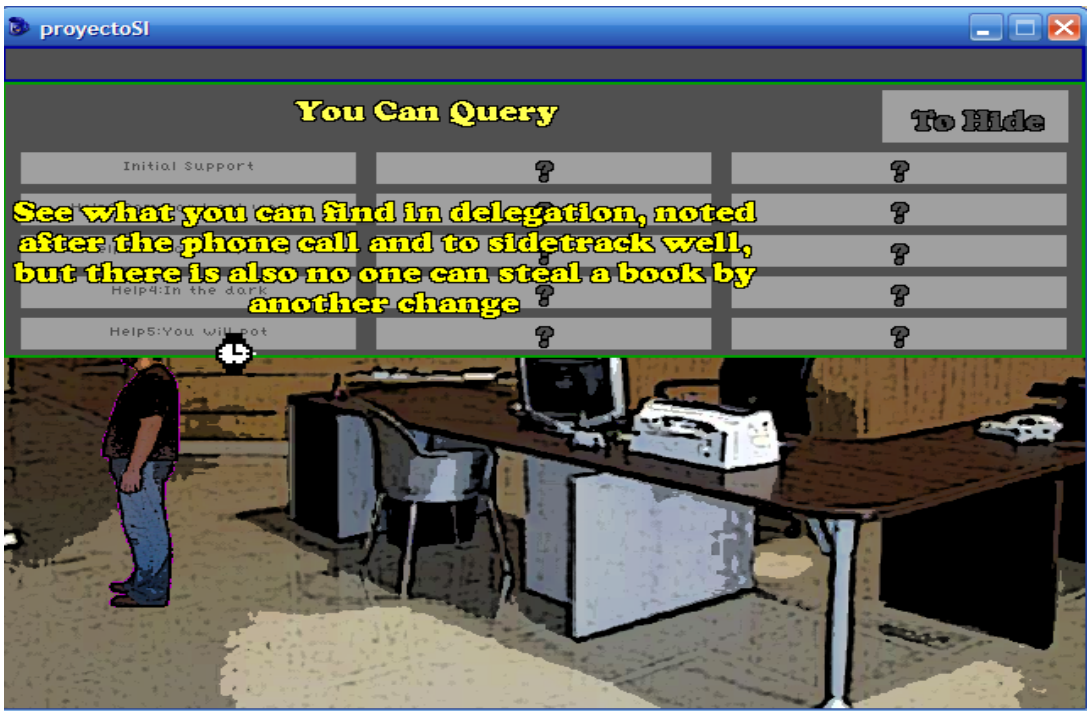


Figura 4.45: Ejemplo traducción Ayuda

4.3. Mapas

Al probar por primera vez el videojuego, había dos puntos que reducían el interés y la atención del jugador. Por un lado, la sensación constante de no saber en qué lugar de Ciudad Universitaria nos encontrábamos a cada momento, y por el otro, no tener un mapa en el que basarse para saber cómo volver a una determinada pantalla.

Resultó obvio que el juego necesitaba un mapa, por lo que primeramente introdujimos un plano de Ciudad Universitaria que te permitía saber en qué facultad estabas en ese momento (Figura 4.46).

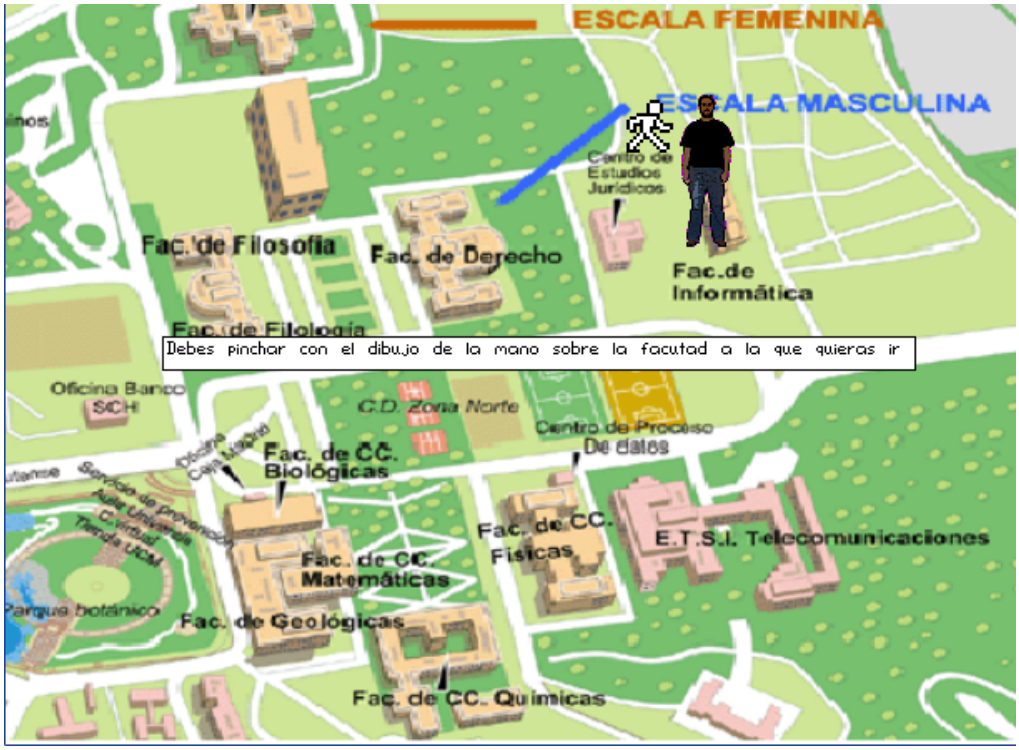


Figura 4.46: Mapa principal

Con esto se solucionaba el tema de la desorientación, pero todavía teníamos problemas a la hora de desplazarnos a un escenario determinado para recoger, por ejemplo, algún objeto que nos hubiésemos pasado por alto.

Decidimos por tanto, implementar algún tipo de 'teletransporte'. La idea inicial fue introducir mapas de cada facultad intentando seguir el emplazamiento de las 'Rooms' en el juego. Sin embargo, fue imposible, debido sobre todo a que este emplazamiento no se correspondía en la mayoría de los casos con el real.

Solventamos el problema mediante la introducción de una nueva Room para cada facultad (Figura 4.47). Estas Rooms serían accesibles pinchando en el mapa sobre el edificio en concreto al que se quiere ir, y actuarían a modo de 'minimapa' de esa Facultad, permitiendo al usuario transportarse inmediatamente a la zona donde desee ir.

En realidad no son mapas propiamente dichos, sino una colección de imágenes en miniatura de cada una de las zonas más importantes de cada Facultad.

La idea es que a simple vista, el jugador tenga claro ya cuál es la zona a la que le interesa acceder. Basta con que pinche sobre ella y será transportado.

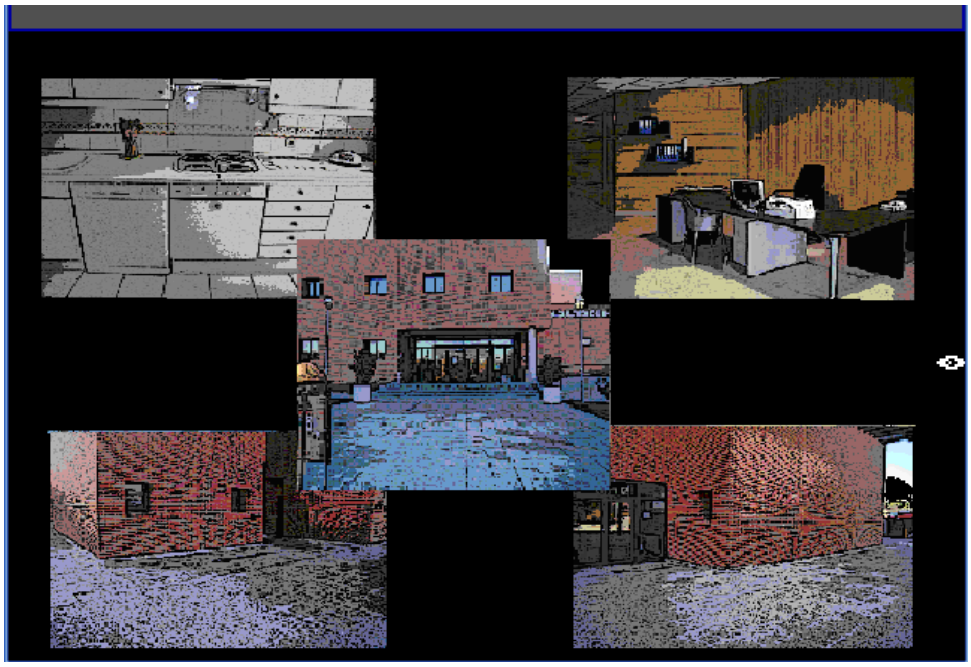


Figura 4.47: Minimaza Informática

Acceso permitido/accedo denegado:

Dado que el juego está implementado por fases, y no todas las facultades son accesibles desde un primer momento, era necesario introducir estas condiciones en los mapas también.

Las reglas que rigen cuándo un mapa es accesible o no, son las siguientes:

FASE	EVENTO	NUEVO VALOR	CONSECUENCIA
0	juego cartas	1	se abre la cocina
1	juego cocina	2	se abre historia
2	juego parejas	3	se abre derecho
3	juego if	4	se abre filología
4	juego while	5	se abre parque de las ciencias
5	recibe sms Eva	6	se abre pasadizo biología
6	entra una vez en pasadizo	7	

Cuando el usuario intenta acceder a una parte del mapa que no está todavía disponible, el juego lanza un mensaje de aviso informándole de que no es posible acceder.

4.4. Tratamiento digital de imágenes

Inicialmente, la aventura gráfica se desarrollaba íntegramente en escenarios 'reales', es decir, sobre fotografías introducidas como pantallas en el juego.

Como ya hemos mencionado anteriormente (véase sección 3.1.1.1 de este mismo documento), la transformación digital de las fotografías en imágenes caricaturizadas de sí mismas supuso importarlas de nuevo una por una al Proyecto del AGS.

Esta transformación dotó al videojuego de una apariencia más amable y atractiva para los usuarios. A continuación mostramos uno de los escenarios antes y después de ser tratado digitalmente con Photoshop.

Al tratarse de fotografías reales (Figura 4.48), el usuario no recibe la sensación de 'estar jugando', resultando menos atractivo.

Después de ser tratadas digitalmente, como se puede apreciar en la Figura 4.49, las líneas se suavizan dando a los escenarios y objetos una imagen 'tipo cómic'.

Figura 4.48: Ejemplo Room antigua

Figura 4.49: Ejemplo Room nueva

4.5. Fases

Consideramos que el juego era muy largo y que se podía hacer algo aburrido para el usuario que estuviese jugando. Por esto, decidimos dividirlo en fases. Pensamos que las fases animarían al usuario a seguir jugando, ya que él mismo se daría cuenta de que iba superando una serie de niveles.

Las fases están colocadas en el juego conforme a los conceptos aprendidos y a los minijuegos superados hasta el momento. De esta manera, conseguimos que el jugador se pueda hacer una idea más clara de las cosas que va aprendiendo, es decir, adquirirá la capacidad de diferenciar los conceptos, a la vez que verá la relación entre ellos. Esto le será de gran utilidad a la hora de superar las pruebas del juego, ya que la mayoría de los minijuegos nuevos introducidos usan conceptos explicados con anterioridad.

A continuación, pasamos a enumerar dichas fases:

- Fase 0: Algoritmos.
- Fase 1: Variables y constantes.
- Fase 2: Asignaciones.
- Fase 3: Tipos.
- Fase 4: Expresiones.
- Fase 5: If-then-else y case.
- Fase 6: Bucle while.
- Fase7: Bucle for.
- Fase 8: Vectores.
- Fase 9: Memoria dinámica.
- Fase 10: Funciones y procedimientos.

Facultad de Informática

Y mostramos un ejemplo de las pantallas de superación de fase que aparecen en el juego (Figura 4.50).

En todas las fases, el jugador tendrá las siguientes opciones:

- Guardar la partida.
- Salir del juego.
- Continuar jugando.

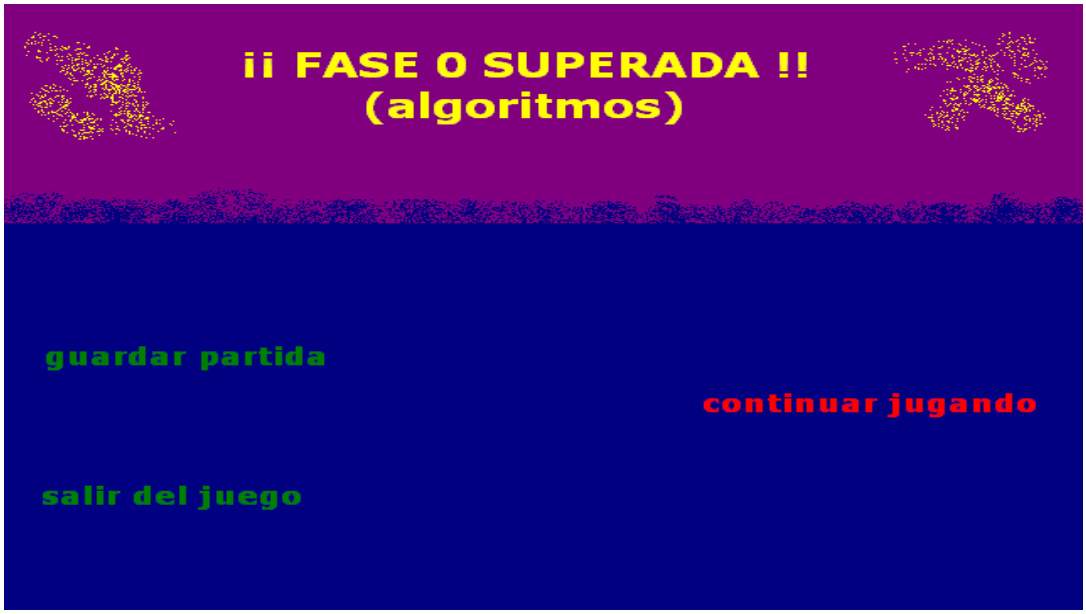


Figura 4.50: Ejemplo Fases

4.6. Ayuda

Debido a la dificultad para superar algunas de las pruebas propuestas para superar el juego, decidimos facilitar al jugador una ayuda que pudiera utilizar en todo momento y que le permitiría obtener algunas observaciones puntuales sobre como superar dichas pruebas.

La ayuda esta asociada a la puntuación, de tal forma que dependiendo de la puntuación que el jugador tenga en un momento concreto tendrá acceso a una parte de la ayuda. Para obtener más puntos y por lo tanto poder así acceder a una ayuda más amplia, el usuario lo único que tiene que hacer es ir superando las pruebas y juegos que se proponen durante la aventura gráfica.

Para que el usuario accediese a esta ayuda hemos facilitado un botón en la barra de herramientas emergente (Figura 4.51), "Ayuda", además junto a este botón hemos introducido la etiqueta que muestra la puntuación obtenida hasta ese momento.

Al pulsar el botón "Ayuda" al jugador se le mostrara otra barra de herramientas (Figura 4.52) en la que aparecerá un botón por cada una de las 15 ayudas que debería obtener durante el juego, de tal forma que si no tiene la puntuación necesaria para acceder a esa ayuda sobre el botón aparece un "?", y si dispone de los puntos correspondientes para tener acceso a ella se muestra un texto resumen de la ayuda a la que se accedería si se pulsa ese botón. Para dejar de ver esta GUI o Barra de herramientas se pulsará el botón "Ocultar". A continuación se muestra la barra de ayuda cuando el jugador dispone de 40 puntos.

En la Figura 4.53 mostramos como aparecería en el juego si se pulsa una ayuda, en este caso la ayuda 3.

Figura 4.51: Barra Menú

Figura 4.52: Barra Ayuda

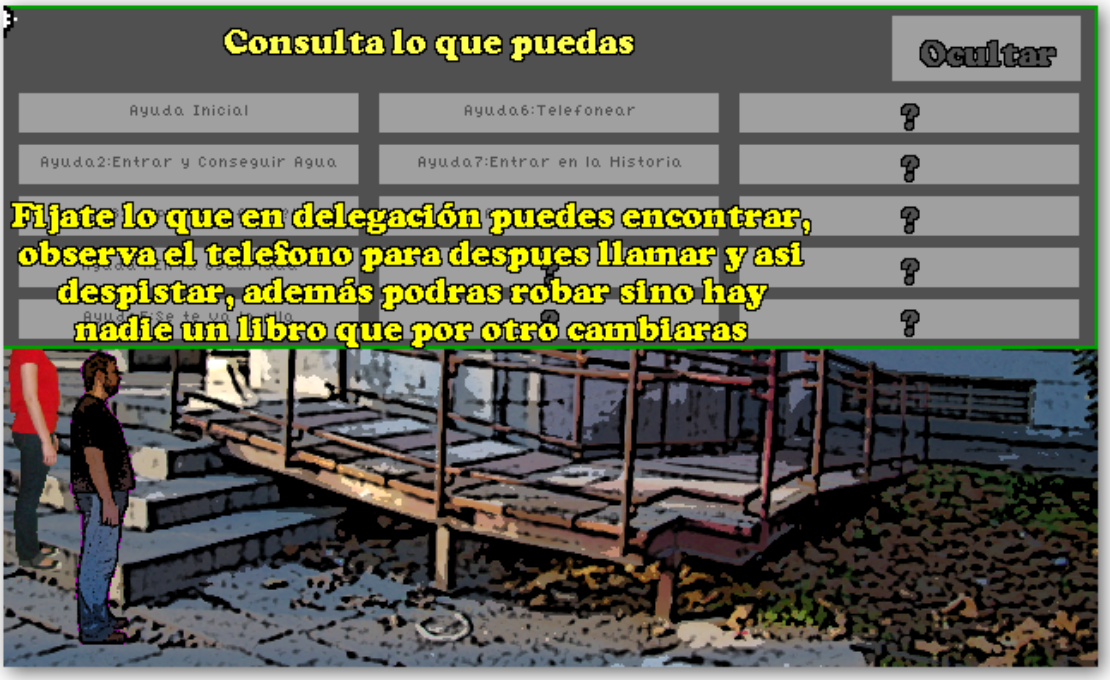


Figura 4.53: Ejemplo Ayuda

Pasamos a detallar toda la información referente a cada una de las ayudas:

→ **Ayuda Inicial** (Figura 4.54)

- **Texto botón:** Ayuda Inicial
- **Cuando: Desde el comienzo**
- **Puntuación: 0**
- **Ayuda: Aparece una nueva habitación en la que se describe la ayuda básica del juego, consistente en el manejo y uso del mismo.**

Figura 4.54: Ayuda Inicial

→ **Ayuda 2**

- **Texto botón:** Entrar y Conseguir Agua
- **Cuando:** Al intentar entrar por la puerta principal de la facultad de informática o al pasar a la zona de la puerta de la cocina
- **Puntuación:** 5

- **Ayuda:** Si agua te piden, detrás de la facultad una boca de riego habrá para que el fuego puedas apagar, además por la puerta de atrás tu entrarás en la facultad

→ Ayuda 3

- **Texto botón:** Despacho Informática
- **Cuando:** Al llenar el vaso de agua en la boca de riego.
- **Puntuación**: 10
- **Ayuda:** Fíjate lo que en delegación puedes encontrar, observa el teléfono para después llamar y así despistar, además podrás robar sino hay nadie un libro que por otro cambiaras

→ Ayuda 4

- **Texto botón:** En la oscuridad
- **Cuando:** Al lograr alcanzar la fase de algoritmo, es decir, después de cocinar el huevo frito
- **Puntuación:** 15
- **Ayuda:** Cerca de unas obras una linterna encontraras que en la oscuridad te alumbrara, cuando algo constante vayas a buscar

→ Ayuda 5

- **Texto botón:** Se te va la olla
- **Cuando:** Al coger por primera vez el autobús hacia historia o al llegar por primera vez a historia
- **Puntuación:** 20

- **Ayuda:** La olla tu necesitaras sino quieres que vapor te salga por las orejas, aunque si estas caliente pidele al borracho algo de beber que con hielo te lo dará

→ Ayuda 6

- **Texto botón:** Telefonear
- **Cuando:** Al realizar el juego de las variables y superarlo
- **Puntuación:** 25
- **Ayuda:** Si a Borja quieres despistar a Delegación de Informática has de llamar

→ Ayuda 7

- **Texto botón:** Entrar en la Historia
- **Cuando:** Al alcanzar la fase de las variables
- **Puntuación:** 30
- **Ayuda:** Si en Historia quieres entrar para a Gonzalo conocer ponte un casco y a trepar para por la ventana meterte, por cierto la fregona has de coger

→ Ayuda 8

- **Texto botón:** Golosa
- **Cuando:** Al superar los juegos que dan la explicación de los tipos enumerados
- **Puntuación:** 35
- **Ayuda:** Carmen que es golosa una chocolatina te pedirá y dentro de derecho la encontraras, por una zona oscura de la explanada accederás

→ Ayuda 9

- **Texto botón:** Que calor!
- **Cuando:** Al superar la fase de las asignaciones
- **Puntuación:** 40
- **Ayuda:** Si a alguien quieres sacar, corta el aire y ya veras como un palo te servirá para algo más que fregar. Te lo digo porque al libro de Carmen quieres llegar y este te abrirá

→ Ayuda 10

- **Texto botón:** Como una estatua
- **Cuando:** Al conseguir llegar a la fase que indica que has entendido el if
- **Puntuación:** 45
- **Ayuda:** El libro que Carmen te pide en filología está y por otro lo cambiaras que había en tu facultad sino lo tienes lo vas a buscar

→ Ayuda 11

- **Texto botón:** A experimentar
- **Cuando:** Tras haberte explicado el concepto del while, al llegar a su fase
- **Puntuación:** 50
- **Ayuda:** Desde el parque de ciencias las siguientes facultades encontraras: Física, Química, Matemáticas y Biología al final

→ Ayuda 12

- **Texto botón:** Encuentra
- **Cuando:** Al coger el autobús hacia el parque de ciencias por primera vez o al acceder desde el mapa a alguna de las siguientes facultades: Física, Química, Matemáticas o Biología

- **Puntuación:** 55
- **Ayuda:** En el parque de ciencias un trapo habrá que te ayudara a limpiar. Además sin una palanca <<química>> el armario del panel, que a alguien ayuda a descender, no abrirás

→ Ayuda 13

- **Texto botón:** A pintar
- **Cuando:** Si has logrado alcanzar la fase del for
- **Puntuación:** 60
- **Ayuda:** Si la brocha quieres untar en el cubo de la pintura que detrás de la valla esta lo conseguirás, pero antes encontrar las llaves deberás en un cuarto que antes del final del pasillo te encontraras. Usa las llaves y el candado se abrirá

→ Ayuda 14

- **Texto botón:** Eva Y Adán
- **Cuando:** Al ganar el juego de los arrays o al haber alcanzado la fase de arrays
- **Puntuación:** 65
- **Ayuda:** Eva en unas escaleras empinadas de historia está y el ascensor arreglaras: 1 - Abre el armario, 3 - Las primeras instrucciones en el panel de derecho buscaras, 4 - Para las segundas instr. en la puerta de Físicas te fijaras, 5 - En el panel el boton1 y ya esta. La combinación te la acabo de aclarar. El final ya no te lo voy a contar, BYE

→ Ayuda 15

- **Texto botón:** Se acerca el final

- **Cuando:** Si has conseguido arreglar el ascensor
- **Puntuación:** 70
- **Ayuda:** Solo era para despedirme y recordar que biología después de matemáticas esta y a Borja lo encontraras en una puerta que lleva a un lugar oscuro. Espero haberte ayudado

4.7. El Gran Libro de la Programación

El objetivo principal que se pretendía conseguir con este juego es que aquellos que quieren aprender programación lo hicieran de una forma más práctica y entretenida, por lo que un punto muy importante es que el jugador pueda consultar en cualquier momento lo aprendido. Pues bien, a medida que vas jugando se van proporcionando conceptos o enseñanzas, según se van superando las pruebas, los personajes o minijuegos enseñan al jugador conocimientos sobre programación. El problema que contemplábamos era que una vez explicado el jugador podía no haber comprendido bien el concepto nuevo, o dicho de otra forma quería consultar lo que le habían enseñado. Por este motivo nuestros compañeros pensaron que sería conveniente facilitarle al jugador una herramienta de consulta durante el juego para que pudiera recordar conceptos adquiridos, y que cuando ellos querían buscar una definición lo hacían en un libro, por lo que crearon un libro que fuera mostrando todo lo aprendido. Decimos todo esto ya que los conceptos enseñados durante el juego los vamos a definir en la memoria igual que se hace en el juego, es decir, utilizando el libro del juego.

El libro se denomina "Gran libro de la programación" y se añade al inventario, a la mochila, en el momento que adquiere el primer conocimiento sobre programación, el concepto de "Algoritmo" (adquirido tras cocinar el huevo). Para

consultarlo sólo tendremos que acceder a la mochila mientras se juega y hacer clic en el botón derecho del ratón situándonos sobre el libro.

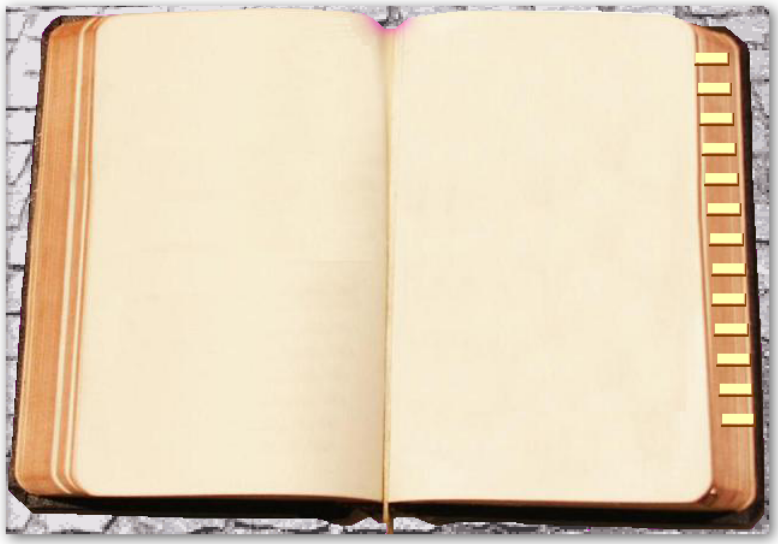


Figura 4.55: Gran Libro de la Programación

En la Figura 4.55 se muestra el aspecto inicial del Gran Libro de la Programación, sobre el que irán apareciendo más páginas cuyo contenido es el concepto enseñado. Para pasar de una explicación a otra solamente tenemos que utilizar las marcas de libro que se observan a la derecha del mismo.

Para poder mostrar la imagen con la explicación adecuada hemos utilizado el objeto "Libro", incluido dentro del apartado "View" del AGS, que permite tener distintas vistas de un mismo objeto. Es como un arrays con imágenes, en donde dependiendo del índice al que accedes se muestra una imagen u otra.

El juego en tiempo de ejecución sabe qué explicaciones del libro puede consultar el jugador, que coinciden con los conocimientos que hasta ese momento ha

adquirido, gracias a la utilización de una variable de estado que hemos denominado "estadolibro".

A continuación se listan los términos sobre programación que durante el juego se enseñan a los jugadores y en el mismo orden en el que aparecen, indicando cuáles de ellos se realizaron en el primer grupo del proyecto y cuáles en el segundo grupo, aunque todos los traducidos al inglés son novedades introducidas este año. También indicamos cuál es el valor que tiene que tener al menos la variable global "estadolibro" para que se pueda mostrar el concepto:

CONCEPTO ---------- GRUPO ----------ESTADOLIBRO

→ Algoritmo. ----------I Grupo ----------1
→ Constantes y Variables. ----------I Grupo ----------2
→ Tipos de Datos. ----------I Grupo ----------3
→ Tipos Enumerados---------- II Grupo ----------4
→ Asignaciones. ----------I Grupo ----------5
→ Expresiones. ---------- II Grupo ----------6
→ Sentencia If. ----------I Grupo ----------7
→ Sentencia Case y tipos subrango. II Grupo----------8
→ Sentencia While. ----------I Grupo ----------9
→ Sentencia For. ---------- II Grupo----------10
→ Vectores. ----------I Grupo ----------11
→ Punteros, Listas, Pilas y Colas. II Grupo----------12
→ Procedimientos y Funciones. -- II Grupo----------13

Ahora expondremos los nuevos conceptos añadidos este año al juego, en el mismo orden utilizado en la lista anterior, y lo haremos mostrando las distintas vistas que tendremos del libro. Todas estas páginas se podrán ver en el libro con el siguiente aspecto:

→ TIPOS ENUMERADOS

"Son tipos que puedes crear y definir tú mismo, es decir, le podemos asignar nosotros mismos los valores que pueden tomar.

Debemos definirlos en una nueva sección llamada "TYPE" y luego declarar una variable del tipo que has definido para poder usarlo.

Se escribiría así:

```
Type
    nombre = (valor1,…, valorn);
Var
    variable : nombre;
```

Ejemplo: si la variable color pudiera tener solo los valores rojo, verde y azul.

```
Type
    colores = (rojo, azul, verde);
Var
    color : colores;"
```

→ SENTENCIA CASE Y TIPOS SUBRANGO

"La sentencia Case surge como alternativa a las sentencias If anidadas, y por lo tanto se usa para elegir una y solo una entre distintas alternativas.

CASE Variable OF

[Valor1Var: Sentencias si valor1;]

[Valor2Var: Sentencias si valor2;]

…

[ValorNVar: Sentencias si valorN;]

[ELSE Sentencias si ninguno de los valores anteriores]

;

Valor puede ser una constante o un subrango de valores, que tendrán el mismo tipo de dato que Variable. Si Variable tiene el Valor1Var, entonces se ejecutarán las sentencias de Valor1Var.

Un SUBRANGO, se utilizará en la sentencia Case ya que en ella no se pueden poner expresiones de comparación para especificar condiciones. Se escribe:

valorInicio..valorFin

Por ejemplo: para referirme a los números del 1 al 5, escribiría: 1..5

Un ejemplo de utilización:

Nota:=5;

CASE Nota OF

0..4: Writeln('Suspenso.');

5..6: Writeln('Aprobado.');

7..8: Writeln('Notable.');

9: Writeln('Sobresaliente.');

10: Writeln('Matrícula.');

Else writeln('Valor no válido.')

;"

→ PUNTEROS, LISTAS, COLAS Y PILAS

"PUNTEROS: es una variable que guarda la dirección de memoria de otro objeto (ej: de otra variable). Es como una caja cuyo contenido es una flecha que apunta a otra caja y del mismo tipo que la caja a la que apunta.

LISTAS: como un vector pero no tiene un número de elementos prefijado, sino que se crea vacía y luego se van añadiendo elemento (en la posición que queramos). En ocasiones cada elemento tiene dos punteros, uno que apunta al siguiente elemento en la lista y otro al anterior.

PILAS: *LIFO: Last in, First out*, es como una lista, pero en este caso solo puedes añadir y borrar elementos por uno de los extremos (el apuntado por el puntero de inicio de pila).

Ej.: fila de platos sobre una mesa, se añaden por arriba y se sacan también por arriba, el último en entrar es el primero en salir.

COLAS: *FIFO: First in, First out*, es como una flista, pero se van a añadir elementos por un extremo y a borrar por el otro. Tendrá dos punteros, uno al inicio y otro que apuntará al último por el cual se añadirán.

Ej.: fila de personas esperan comprar una entrada de cine, el primero en llegar es el primero que sale."

→ SUBPROGRAMAS: PROCEDIMIENTOS Y FUNCIONES

"PROCEDIMIENTOS

Porción de código dentro de un programa más grande, que realiza una tarea específica y es relativamente independiente del resto de código. Son llamados desde otros procedimientos o funciones, siendo su llamada una línea más de código.

Los procedimientos pueden recibir parámetros, que son como variables enviadas desde el exterior, de entrada, o que emiten al exterior, de salida.

Se diferencian de las funciones en que éstas devuelven un único valor y siempre a través de su propio nombre, mientras que el procedimiento puede devolver varios valores pero a través de parámetros.

FUNCIONES

Grupo de instrucciones con un objetivo en particular y que se ejecuta al ser llamada desde otra función o procedimiento.

Su principal característica es que debe devolver un resultado a través de su propio nombre y por tanto hay que asignarle un tipo de dato que se adapte al resultado. Además al devover un valor ha de ser llamada desde una expresión.

Las funciones pueden recibir parámetros pero éstos han de ser de entrada, es decir, no pueden devolver ningún valor al exterior, solo introducirlo en la función."

5. Guía del usuario

5.1. Inventario

Este tipo de juegos, las aventuras gráficas, tienen la característica de que en ocasiones los diseñadores los desarrollan de tal forma a veces resulta demasiado complicado averiguar la solución para superar determinadas pruebas. De hecho nosotros en cierta medida, reconocemos que algunas pruebas del juego entrañan cierta dificultad para adivinar cómo superarlas, sobre todo en aquellas situaciones en las que no sabemos bien dónde y cuándo es conveniente utilizar alguno de los objetos del inventario.

Pues bien, con el objetivo de describir y, principalmente, facilitar al jugador un resumen de los objetos de inventario, proporcionamos a continuación información detallada de cada uno de los objetos.

5.1.1. Utilización del Inventario

Lo primero es explicar cómo podemos acceder y utilizar los objetos que vayamos consiguiendo. Si mientras estamos jugando nos topamos con un objeto que aparezca superpuesto en el fondo o si al situar el puntero del ratón sobre dicho objeto, nos aparece una descripción en la barra de estado, entonces seguramente hayamos encontrado un nuevo objeto de inventario.

Todos estos objetos se van añadiendo a la mochila que tiene el jugador y normalmente se suelen utilizar en algún momento del juego. Para añadir el objeto a la mochila solo tenemos que poner el puntero del ratón en Mode 2 ó Interact, , y pulsar sobre él, automáticamente se añadirá (Figura 5.1).

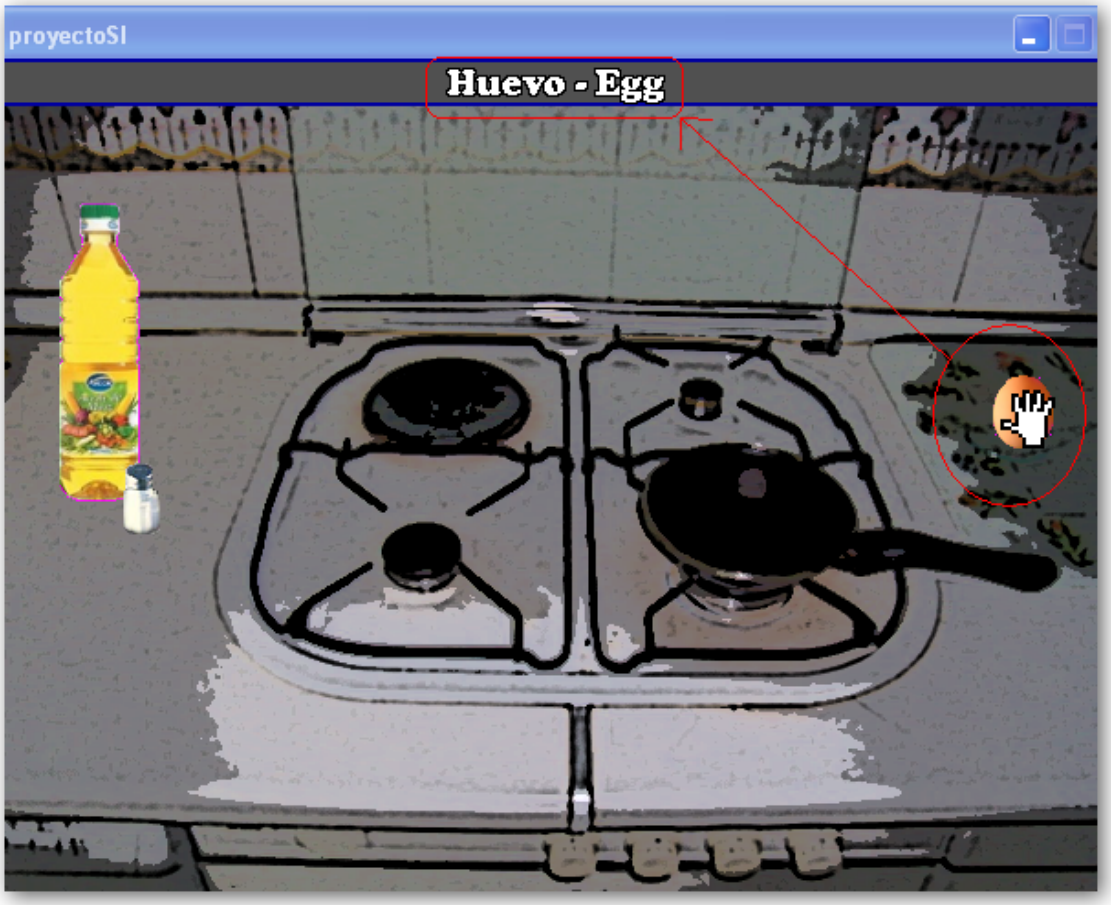


Figura 5.1: Interact

Tenemos que aclarar que hay algunos objetos que no es necesario coger de ningún sitio, sino que en determinadas partes del juego se añaden directamente a la mochila, como por ejemplo si nos los proporciona alguno de los personajes o se intercambia por otro objeto.

Si queremos recuperar alguno de los objetos porque creemos que es necesario utilizarlo en algún momento, tendremos que ir a la barra de herramientas emergente (Figura 4.51) y pinchar sobre la imagen de la mochila.

Facultad de Informática

Nos daría acceso al inventario donde tenemos todos los objetos conseguidos hasta ese momento en el juego y simplemente tendríamos que hacer clic sobre el que queramos utilizar. Podemos saber cuál de los objetos de la mochila es el que tenemos activo en ese momento si aparece en el hueco que en la imagen anterior está vacío y también si una de las apariencias del puntero del ratón es el objeto (Figura 5.2).

Una vez seleccionado el objeto, para utilizarlo solo tenemos que poner el puntero del ratón con su imagen y ponerlo allí donde lo tengamos que utilizar. En la Figura 5.3 mostramos cómo se llenaría el vaso de agua, simplemente poniendo encima de la boca de riego y haciendo clic.

Figura 5.2: Barra Menú con Objeto de Inventario

Figura 5.3: Ejemplo interacción con un objeto

5.1.2. Objetos del Inventario

Ahora ya trataremos cada uno de los objetos del inventario por separado.

→ **VASO** 

Este objeto se lo proporciona Borja a Lucas en el despacho de la Facultad de Informática, añadiéndolo directamente a la mochila, por lo que Lucas no lo tiene que buscar. Lo utilizará en la primera prueba que Borja le mande, que consiste en que le entregue el vaso lleno de agua para saciar su sed.

Para llenar el vaso de agua, Lucas ha de ir a la parte de atrás de la facultad de Informática donde encontrará la boca de riego que se encargará de tal labor. Se puede apreciar en la imagen anterior.

Una vez esté el vaso lleno tendrá esta apariencia,  , y ya se le podrá entregar a Borja que lo estará esperando en el mismo despacho, dándose así la prueba por superada.

→ **Botella de Aceite, Huevo y Sal**   

Englobamos estos tres objetos en el mismo apartado debido a que se utilizan y forman parte de la misma prueba. Los podemos conseguir cuando se nos explica lo que es un algoritmo, ya que precisamente el ejemplo utilizado es el del proceso de creación de un huevo frito.

Facultad de Informática

Nos los proporcionarán para que hagamos uso de ellos en el mismo momento, de forma que tendremos que cocinar un huevo frito en la cocina de la propia Facultad de Informática.

Los pasos a seguir son:

- Verter Aceite
- Encender Fuego
- Echar Huevo
- Sacar Huevo 
- Echar Sal

→ **Libro de la Programación** 

El Gran Libro de la Programación se consigue justo después de cocinar el huevo frito en la cocina de Informática. Nos lo dará Borja tras explicarnos el concepto de Algoritmo. Este libro sirve como consulta de los conceptos que según va transcurriendo el juego Lucas va adquiriendo, de tal forma que inicialmente solo podemos ver la explicación de Algoritmo y posteriormente, según avanzamos, el jugador podrá acceder a más secciones.

Se puede consultar en cualquier momento una vez conseguido, para ello simplemente tenemos que acceder a la mochila y pulsar el botón derecho sobre este libro.

→ **Bolsa Isotérmica y Hielo**  

La bolsa la podemos coger en la cocina de la facultad de Informática, se encuentra sobre la encimera. El hielo nos lo proporcionará el borracho que se encuentra en la entrada trasera de la facultad de Informática, pero sólo cuando estemos en la fase de las variables, y después de insistirle mucho.

La vamos a necesitar en la prueba en la que Carmen explica el concepto de las variables, donde se pone como ejemplo el agua que puede estar en tres estados diferentes. Uno de esos estados es el sólido, hielo, éste no se lo podemos llevar a Carmen como tal, porque se deshace antes de entregárselo, por lo tanto lo tendremos que meter dentro de la bolsa isotérmica. Para meter el hielo en la bolsa deberemos acceder a la mochila y arrastrar el hielo sobre la bolsa o viceversa.

Una vez introducido el hielo en la bolsa se derrite con menos facilidad y se le podrá entregar a Carmen, la bolsa.

→ **Linterna** 

El objeto linterna lo podemos encontrar en una zona en obras que hay en la facultad de Historia, concretamente tendremos que observar en un montón de arena que hay para poderla conseguir, ya que no se ve a simple vista (Figura 5.4).

Como se puede apreciar también podemos encontrarnos aquí el objeto "casco".

La linterna nos será útil allí donde no veamos bien. Concretamente la tendremos que utilizar durante la prueba de las constantes, ya que a la hora de ir a buscar algo que permanezca invariable mucho tiempo, el colgante, tendremos que mirar dentro de una alcantarilla que se encuentra en la entrada secundaria de la facultad de Historia. Obviamente la alcantarilla está muy oscura, por lo que sin la linterna no seremos capaces de ver nada.

También podríamos utilizar más adelante este objeto cuando entremos en una habitación a oscuras de la facultad de Biología, pero el jugador sufrirá un desgraciado accidente y la linterna se estropeará.

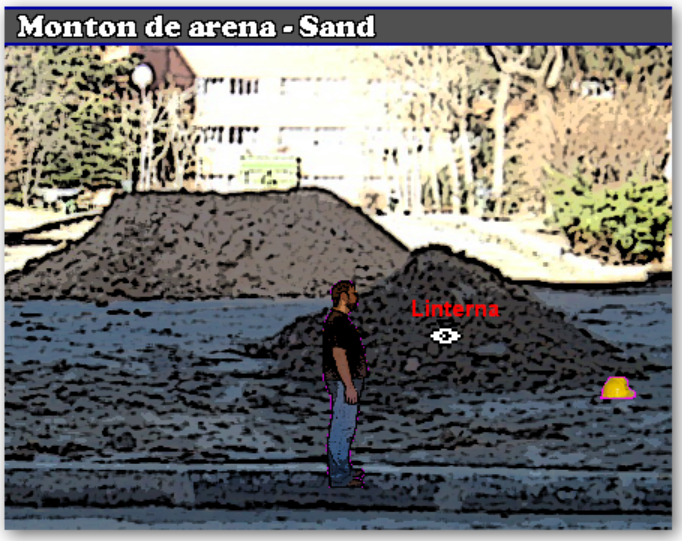


Figura 5.4: Ubicación linterna

→ **Casco** 

El casco de obra lo podemos encontrar en la zona en obras de la facultad de historia, como se puede ver en la imagen en la que se muestra el objeto linterna.

Este casco lo podemos utilizar para que a Lucas le dé la seguridad y valor suficientes para subir por el árbol que se encuentra en la zona del patio, y así entrar por la ventana que le da acceso al interior de la facultad de Historia, donde se encuentra Gonzalo. Esto lo tendrá que hacer después de la fase de las variables, cuando Carmen le pida a Lucas que entre en Historia y busque a Gonzalo, quien tiene un libro que ella le había prestado y quiere recuperar.

→ Colgante 

Forma parte de la prueba de las constantes, en la que Carmen le pide a Lucas que le entregue algo que haya permanecido durante mucho tiempo invariable.

Lo podemos encontrar dentro de una alcantarilla, como hemos mencionado anteriormente, que se encuentra en la puerta secundaria de la facultad de Historia (Figura 5.5). Una vez que se lo entreguemos a Carmen, habremos superado la prueba de las constantes. Por el camino nos encontraremos a una chica que parece nerviosa buscando algo. Deberemos hablar con ella para que nos diga qué es lo que busca.

Figura 5.5: Alcantarilla

→ **Olla** 

La necesitamos para superar la fase de las variables, ya que Carmen nos pide que le entreguemos el agua en los tres estados: sólido, líquido y gaseoso. El líquido ya lo tenemos, ya que Borja nos pidió agua en un vaso. El estado sólido lo conseguimos tras insistirle mucho al borracho que se encuentra en la facultad de Informática para que nos dé hielos del mini que tiene en la mano (importante acordarse de meter el hielo en la bolsa isotérmica). El estado gaseoso suponemos que lo obtienes de la olla a presión que está utilizando Borja para hacerse un cocido.

La olla a presión, como hemos indicado, la está utilizando Borja en la cocina de la facultad de Informática, pero solo estará disponible a partir del momento en que Carmen nos pida los tres estados del agua. Lo que está claro es que no se la podemos coger a Borja delante de sus narices, por lo que le tendremos que sacarle de la cocina. Para ello, tendremos que despistarle llamando al teléfono del despacho de delegación de Informática desde la cabina que se encuentra en la facultad de Historia. Como no nos sabemos el número de teléfono de delegación deberíamos buscarlo en la propia delegación. Una vez conseguido, podemos llamar y hacernos pasar por el decano de matemáticas y tras mandarle a hacer un recado, Borja saldrá y podremos coger la olla libremente.

→ **Fregona** 

La debemos coger dentro de la facultad de Historia, allí donde nos encontramos con Gonzalo. El momento propicio para utilizarla será en la facultad de Filología. En un momento determinado del juego, deberemos pasar una verja que está cerrada y da acceso a la estatua donde está el libro que necesita. Para poder abrir

dicha verja, tendremos que hacer salir al bedel estropeando el aire acondicionado (Figura 5.6) para hacer que salga y nos abra la puerta. La manera de estropear el aparato es introduciendo la fregona.

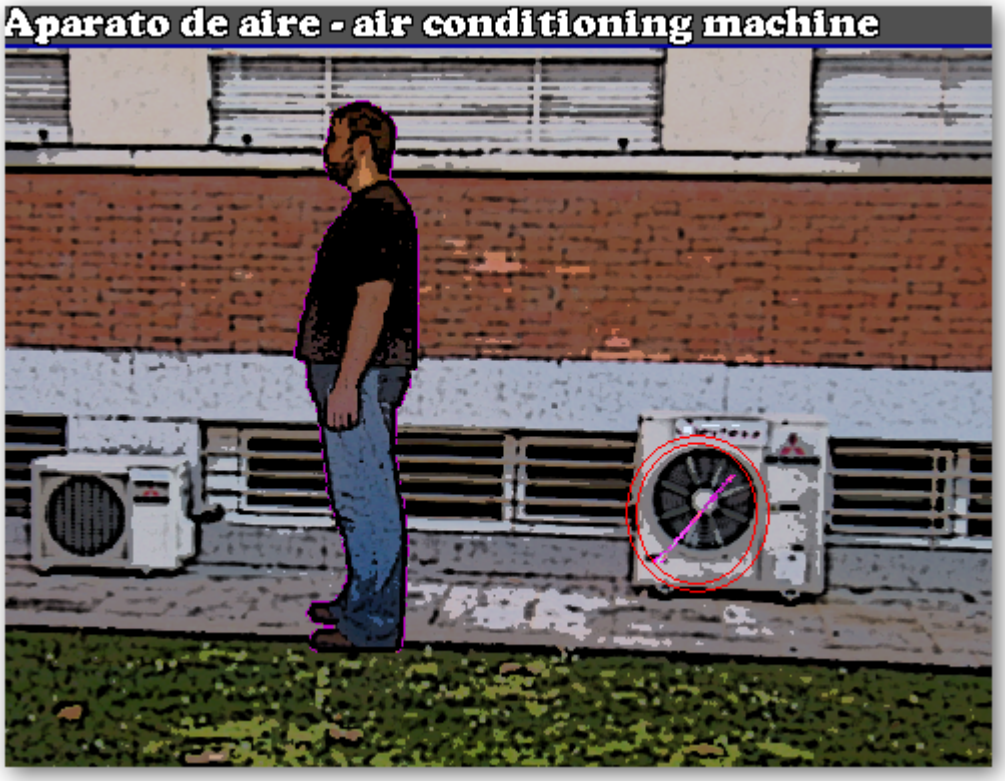


Figura 5.6: Aparato de aire acondicionado

→ Libro: Alicia en el País de los bits

Este libro se encuentra en el despacho de delegación en la facultad de Informática, en una estantería (Figura 5.7). Para poder cogerlo es necesario que Borja no se encuentre en el despacho. Recomendamos que se obtenga antes de llegar al laberinto que se encuentra en Filología.

Lo utilizaremos para intercambiarlo por el libro que está buscando Carmen, para que la estatua no llame la atención sin su libro. Sería conveniente tenerlo antes de que se cruce el laberinto que da acceso a la estatua.

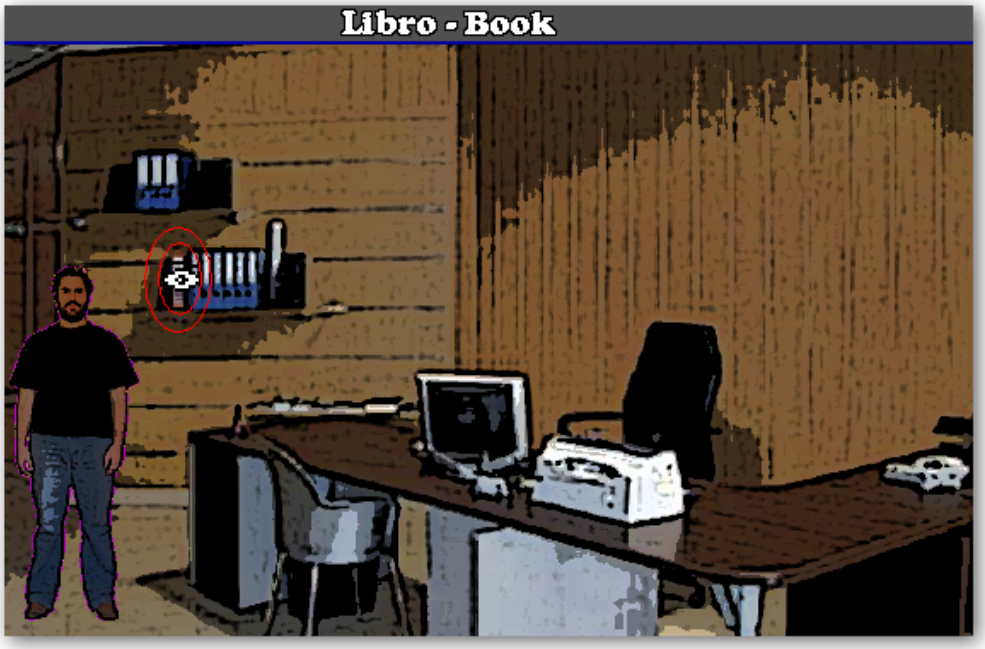


Figura 5.7: Libro del despacho de Delegación

→ **Chocolatina** 

Para explicarnos las asignaciones estando en la facultad de Derecho, Carmen nos dará un euro,  , y le pedirá a Lucas que le consiga una chocolatina. Se podrá encontrar en una maquina que se encuentra en una zona a la que se accede por la puerta secundaria de la facultad de Derecho.

Facultad de Informática

Tras arrastrar el euro hasta la maquina, la chocolatina aparecerá en la mochila y al volver a hablar con Carmen nos dará por pasada la prueba y nos explicará el concepto de las asignaciones.

→ Libro: Carmen. 

Estamos hablando del libro que Carmen primero dijo a Lucas que pidiera Gonzalo y luego resultó que lo había dejado olvidado en la estatua de Ortega y Gasset. A esa estatua tendremos que ir a buscarlo, pero no lo podremos coger si no tenemos otro libro por el cual sustituirlo (Alicia en el País de los Bits).

→ Palanca 

La palanca la tenemos que ir a buscar a la puerta de la Facultad de Químicas, estará apoyada sobre un coche (Figura 5.8).

Figura 5.8: Palanca coche

La tendremos que utilizar para abrir el armario que contiene en su interior el panel del ascensor por el que tenemos que conseguir que baje Eva. El ascensor está localizado en el nuevo edificio.

→ **Brocha** 

Cuando nos volvemos a encontrar con Carmen en Ciencias después de superar el minijuego del "For", nos dará una brocha que nosotros tenemos que untar de pintura.

El bote de pintura donde tenemos que meter la brocha se encuentra en la Facultad de Matemáticas detrás de una valla cerrada con candado, por lo que no podemos acceder a la pintura hasta que no encontremos las llaves que abren dicho candado.

Las llaves del candado,  , las podemos conseguir en un almacén a oscuras (Figura 5.9) en el que Lucas sufrirá un pequeño percance mientras las busca. Esa habitación está en la Facultad de Biología.

Una vez con las llaves en nuestro poder, ya podemos ir a la valla, abrir el candado y, una vez dentro, untar la brocha que ahora pasará a tener pintura,  .

Con la brocha llena de pintura volveremos donde habíamos dejado a Carmen, quien nos pedirá que pintemos en los cubos para explicarnos lo que son los arrays.

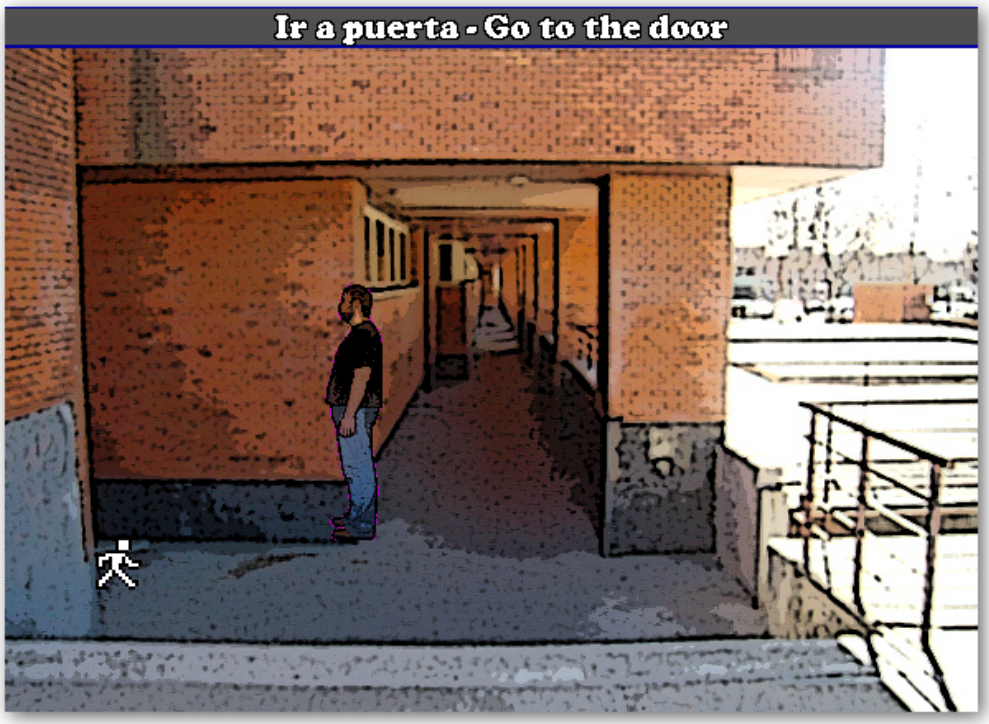


Figura 5.9: Acceso al almacén oscuro

→ **Trapo** 

Tras haber escrito en los cubos, Carmen ahora le dice a Lucas que limpie lo que ha escrito. Con este fin, el de limpiar los cubos, Lucas ha de buscar un trapo.

El trapo se encuentra cerca del distribuidor de ciencias, metido en un agujero. Una vez en sus manos, Lucas ha de volver con el trapo y limpiar los cubos (Figura 5.10).

Figura 5.10: Cómo usar el trapo con los cubos

→ **Instrucciones**

Carmen ha enviado a Lucas a que ayude a Eva, que se encuentra en el edificio nuevo y no puede bajar por las escaleras ya que se encuentra lisiada (razón por la cual Lucas deberá arreglar el ascensor),

No sirve con haber conseguido abrir el panel eléctrico de los interruptores del ascensor con la palanca, además Lucas ha de conseguir las instrucciones para arreglarlo. Esas instrucciones están en:

- Las primeras en el tablón de anuncios de la Facultad de Derecho
- Las otras en la puerta de la facultad de Físicas.

La combinación de botones del panel del ascensor es levantar los interruptores 1, 3, 4 y 5 y después el botón 1.

5.2. Guía de usuario: cómo jugar

En este capítulo se enseña a un usuario sin experiencia en juegos de tipo aventura gráfica cómo dar sus primeros pasos. En general, consideramos que es bastante intuitivo, sin embargo si el jugador nunca se ha enfrentado a este tipo de interfaz, será de agradecer una explicación detallada como esta.

5.2.1 Seleccionando la acción

En el juego disponemos de 4 acciones básicas:

- Ir a.
- Mirar.
- Hablar a.
- Coger/usar.

Tenemos dos formas de seleccionar una opción:

1. Pulsando el botón derecho del ratón repetidamente hasta que el cursor coincida con la acción que queremos realizar.
2. Seleccionando en la barra emergente. Ésta se oculta automáticamente. Para visualizarla, basta con pasar el puntero del ratón por la zona superior de la pantalla. (Véase Figura 5.11 y Figura 5.12).

Los botones de la barra (véase Figura 5.13), ordenados de izquierda a derecha son los siguientes:

1. Andar.
2. Mirar.
3. Coger/Usar.
4. Hablar.

5. Abrir inventario.
6. Objeto seleccionado del inventario (en gris si no hay ninguno seleccionado).
7. Salvar la partida.
8. Cargar la partida.
9. Salir del juego.
10. Información sobre el juego.
11. Acceso a mapas.
12. Puntuación.
13. Ayuda.

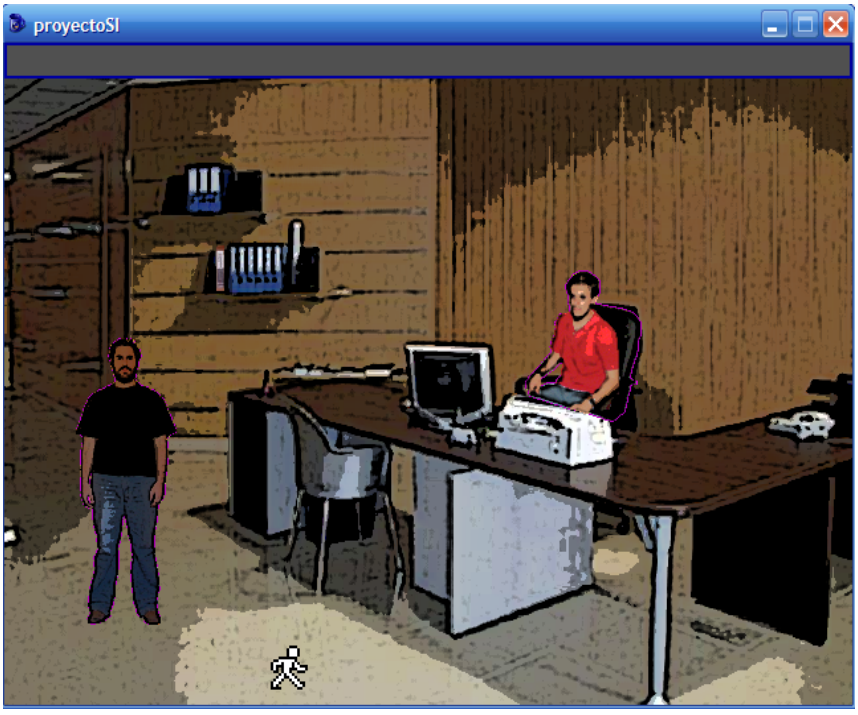


Figura 5.11: Barra oculta

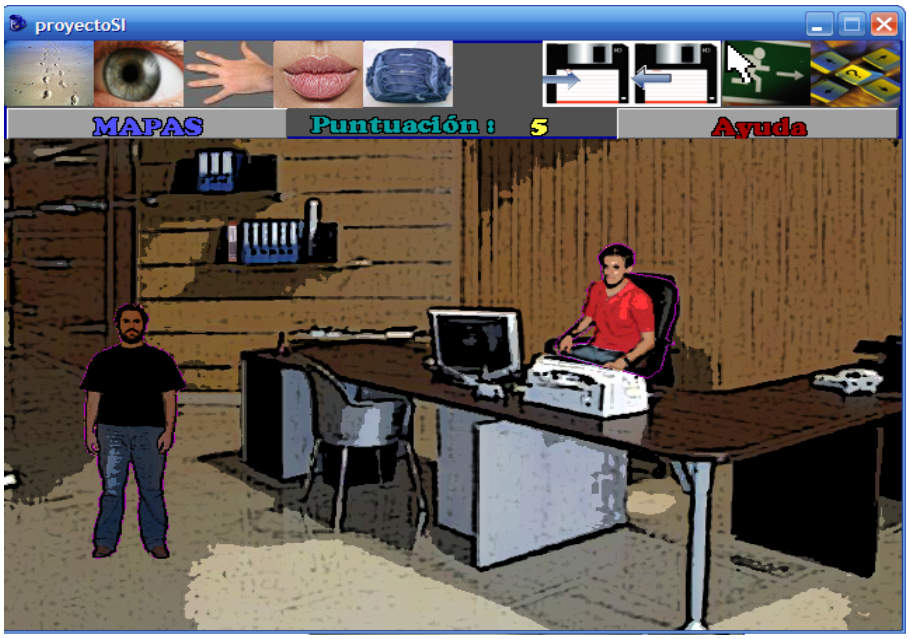


Figura 5.12: Barra visible

Figura 5.13: Barra menú

5.2.2 Inventario

Para acceder al inventario tendremos que seleccionar la mochila que hay en la barra emergente. Entonces se abrirá una ventana en la que podemos ver todos los objetos que tenemos disponibles. Cada objeto del inventario puede ser inspeccionado o seleccionado. Para mirarlo tendremos que pulsar sobre el icono de ojo que hay dentro del menú del inventario. Si por el contrario lo que queremos es seleccionar un objeto para usarlo fuera del inventario, tendremos que hacer clic sobre el objeto teniendo el cursor normal activo, y una vez que tengamos el objeto en cuestión como

cursor, pulsar al OK. De esta manera se cerrará el menú de inventario y el objeto estará preparado para ser utilizado en el escenario.

5.2.3 Las acciones

1. Ir a: lo utilizaremos para que el personaje se mueva por las habitaciones y para cambiar de habitaciones. Cuando entremos en ciertas zonas de las habitaciones, haremos que el personaje cambie de habitación (generalmente son los extremos de las habitaciones, y las puertas).
2. Mirar: Esta acción nos dará generalmente más información sobre un objeto, persona o parte del escenario que no vemos a simple vista y que puede resultarnos útil en el desarrollo del juego. Por ejemplo, podemos ver que hay una boca de incendios en el escenario, pero sólo si seleccionamos la acción de mirar, y pulsamos sobre la boca de incendios, nos daremos cuenta de que esta gotea, de manera que si utilizamos un vaso sobre la boca de incendios, podríamos llenarlo.
3. Hablar a: Generalmente utilizaremos esta acción con personas para comenzar a desarrollar una conversación con los distintos personajes del juego. En contadas ocasiones, podemos hablar con objetos o animales (por ejemplo, si hubiera un loro, tendría sentido intentar hablar con él).
4. Coger/usar: Esta acción puede realizar indiferentemente dos cosas:
    a. Coger: Si teniendo seleccionada esta acción pulsamos sobre un objeto, el protagonista lo cogerá y lo añadirá a su inventario en el caso de que este sea interesante. Por ejemplo, si hay un salero y lo seleccionamos, se añadirá a nuestro inventario y desaparecerá del escenario.
    b. Usar tiene 2 vertientes:
        I. Un objeto del escenario puede tener una función por si mismo, por ejemplo un interruptor. Si lo seleccionamos, probablemente

se encienda la luz. Para esto bastará con tener seleccionada la acción de usar, y pulsar sobre el objeto interruptor.

II. Usar con: A diferencia de las demás acciones esta no tiene su propio icono en la barra de herramientas ni en la serie de iconos que aparecen pulsando el botón derecho del ratón. Para realizar este tipo de acciones tendremos que seleccionar el objeto del inventario, pulsar sobre el botón de OK, y posteriormente utilizarlo sobre lo que queramos. Un objeto puede usarse sobre:

1. Otro objeto. Por ejemplo, si utilizamos una moneda de euro con una máquina conseguiremos un snack que será añadido a nuestro inventario. Para realizar esta acción tendremos que seleccionar del inventario la moneda de euro; una vez que esta aparezca en el cursor, podremos moverla hacia el objeto con el que queramos interaccionar y pulsaremos sobre él.
2. Un personaje: De este modo le intentaremos dar el objeto al personaje seleccionado.
3. El propio jugador: Por ejemplo, si queremos que el protagonista se ponga un casco, tendremos que seleccionar el casco en el inventario y pulsaremos sobre el propio personaje.

5.2.4 Cómo cambiar de habitación

En el desarrollo del juego, el personaje deberá cambiar de una habitación a otra. Para ello el jugador deberá seleccionar la acción andar y seleccionar en una zona de la habitación. Si la zona seleccionada no es accesible, el personaje se acercará lo máximo posible a ese punto, siempre manteniéndose dentro de la zona *walkable*.

Normalmente, las zonas de transición están marcadas de manera que al pasar el ratón por encima saldrá algún nombre en la barra superior. Normalmente estas zonas son los extremos de las habitaciones y las puertas.

En el desarrollo del juego, hay algunas zonas que pueden estar bloqueadas hasta que no se realice una acción determinada. También hay algunas zonas a las que se accede realizando alguna acción. Por ejemplo, si hay una puerta y al intentar pasar se dice que está cerrada, el personaje deberá usar algún objeto para abrir la puerta y pasar así a la nueva zona o habitación.

Además, desde el comienzo del juego el jugador tendrá acceso al mapa de Ciudad Universitaria, a través del cuál podrá consultar en todo momento dónde está, y transportarse automáticamente a la zona deseada (siempre que dicha zona sea accesible, dependiendo de en qué fase del juego se encuentre). Para ello, deberá pulsar la opción del menú correspondiente, MAPAS, que le redigirá al mapa principal.

Una vez allí, el usuario deberá seleccionar la Facultad donde quiere ir con cualquier botón de acción MENOS el de andar. Le aparecerá por pantalla un mosaico con imágenes en miniatura de las áreas principales de dicha Facultad. Bastará con que pulse con cualquier botón de acción sobre una de ellas, y Lucas aparecerá automáticamente en dicho escenario.

5.3. Guía del juego

Lucas A. es un futuro alumno de Ingeniería Informática en la UCM. A finales de agosto, Lucas se acerca a su futura facultad para informarse de la fecha del acto de

bienvenida para los alumnos de primer curso. Al llegar encuentra que todo está bastante vacío en la zona de la entrada de la facultad, por los alrededores sólo ve un chico bebiendo un mini, sentado a la sombra.

Lucas sigue paseando un poco y sólo encuentra abierta la delegación de alumnos, donde hay un chico en un ordenador que parece ignorarle. Cuando Lucas habla con él, este tendrá una amable conversación. Bromeará con Lucas y le dirá que si quiere ser informático tendrá que engordar unos cuantos kilos, tener siempre una coca cola a mano, y llevar camisetas frikies (tras esto emite una estruendosa risa).

Seguidamente, y en un tono más serio, le dice a Lucas que lo que sí que necesitará realmente será aprender a programar y le preguntará si quiere aprender. Lucas responde que le da pereza, que está de vacaciones. Sin embargo parece que el chico de delegación (a partir de ahora, Borja) tiene mucho interés en que Lucas aprenda a programar, puesto que necesita que alguien pruebe el juego tipo gymkhana que ha creado para enseñar los conceptos básicos de la programación.

Borja enseña a Lucas un abono transporte, que parece ser que Lucas ha perdido por ahí. Borja le dice a Lucas que si quieren se lo apuestan. Borja enseñará tres cartas a Lucas y las moverá. Si Lucas acierta dónde esta el rey, Borja le devolverá el abono, si no, Lucas tendrá que probar el juego y aprender a programar. Lucas pierde, y aquí es donde empieza el juego de verdad.

En primer lugar Borja parece que tiene sed, así que manda a Lucas a llenar un vaso de agua. Para llenarlo, Lucas lo podrá utilizar una boca de incendios que hay cerca de la puerta de delegación.

Una vez entregado el vaso a Borja (que, para fastidiar, dirá que no lo quiere), éste le empezará a explicar el concepto de algoritmo.

Borja le dirá a Lucas cinco tareas para conseguir un huevo frito, y este tendrá que decirlas en el siguiente orden:

- Echar aceite
- Encender el fuego
- Echar el huevo
- Echar sal
- Sacar el huevo.

Cuando Lucas lo completa correctamente Borja le dirá que muy bien hecho, y le lleva a la cocina, (cosas que tiene ser de delegación...). A partir de este momento Lucas podrá entrar a la cocina, que hasta ahora estaba cerrada. Podrá acceder desde la puerta exterior, a la izquierda de la entrada principal.

En la cocina Lucas tiene que realizar la tarea de freír un huevo tal y como lo hizo en la conversación con Borja. Hasta que no lo acabe no podrá salir de la cocina. Se encontrará un mostrador con los ingredientes necesarios, y la sartén encima de la cocina. Pulsando el botón de la derecha se encenderá el fuego.

Cuando lo complete, Borja le dará "El gran libro de la programación", en el cual, a partir de ahora podrá apuntar los conocimientos que vaya adquiriendo sobre programación. Tras echar un vistazo al libro y a la explicación de Algoritmo, Borja le pedirá un favor a Lucas. Necesita que este vaya a Historia a buscar a su novia Carmen para decirle que no podrá acudir a una cita. Lucas podrá ir a historia en autobús, cogiéndolo en la puerta de informática.

En la entrada lateral de derecho se encuentra con una chica que resulta ser Carmen. Tras dialogar un poco con ella, parece que Carmen le va a enseñar algunas cosas sobre programación. En primer lugar le pide que busque algo constante. Cuando empieza la búsqueda, Lucas puede hablar con una chica que parece nerviosa. Si lo hace ella le dirá que busca un colgante que tiene mucho tiempo. Esto es una pista para que Lucas sepa que por la zona debe haber algo que lleva mucho tiempo sin cambiar, porque es muy antiguo.

El colgante resulta estar en una alcantarilla (que hay que abrir) en la facultad de historia. Para poder verlo hay que iluminar la alcantarilla utilizando una linterna que se encuentra en unas obras cerca de historia. Para encontrar la linterna hay que mirar dentro de un montón de arena.

Una vez conseguido el colgante, Lucas se lo debe dar a Carmen y ella le hará una breve explicación de lo que es una variable, y le pedirá que consiga algo que tenga varios estados, concretamente agua. Lucas ya tiene un vaso de agua, por lo que sólo le quedarán los estados sólido y gaseoso.

- **gaseoso:** Borja se está haciendo unos espaguetis en la cocina. Hay que llamarle (desde la cabina) al despacho para que se vaya de la cocina y poder coger la olla. (para tener el número, hay que haber mirado el teléfono del despacho). Aquí Lucas se hace pasar por el rector y le hará un encargo, sacando así a Borja de la cocina y del despacho de delegación.
- **hielo**: El chico que está fuera de la facultad de informática con un mini, si le pides muchas veces por favor que te de un hielo al final te lo dará.

Sin embargo, si Lucas intenta llevárselo así a Carmen, el hielo llegará derretido y tendrá que volver a pedírselo al chico del mini. La manera de conseguir llevarlo será metiéndolo en una bolsa isotérmica que hay en la cocina.

Una vez que Lucas le entrega las tres variables a Carmen, ésta le explica el concepto de variable, a continuación le comenta que necesita un libro que le ha dejado a su amigo Gonzalo. Carmen se tiene que ir a hacer unas cosas, por lo que quedan en Derecho para cuando Lucas consiga el libro. Al preguntarle Lucas a Carmen dónde está Gonzalo, ésta le contesta que está segura de que llegará a lo más alto. Gonzalo se encuentra en un despacho de la facultad de historia. Como es agosto, la facultad está cerrada, así que para entrar Lucas se tendrá que colar por una ventana trepando por un árbol. Pero a Lucas le da un poco de miedo, por lo que tendrá que escalar usando un casco que encontrará en la obra de cerca de historia. Lucas se dará cuenta de que siente algo especial por Carmen, así que termina poniéndose el casco y subiendo para ayudarla.

Una vez dentro de historia Lucas se encuentra en un pasillo con una zona fregada, por lo que no puede subir donde podrá coger una fregona. También encuentra a Gonzalo, que le explica el concepto de Tipos por medio de un *mini-juego* de hacer parejas. Lucas tiene que unir dos ejemplos de los cuatro tipos diferentes que le explica Gonzalo, Enteros, Reales, Booleanos y Caracteres. Una vez explicado el juego y tomados los pertinentes apuntes, Gonzalo le dice que no tiene el libro de Carmen, que no llegó a dárselo. Lucas podrá ir a Derecho al encuentro de Carmen aunque sin el libro.

Tras la explicación de variables se le hará una breve introducción al concepto de tipos y hará el mini-juego. Una vez superado esta prueba, se le explicará a Lucas lo que es un tipo enumerado y tendrá que superar otro juego.

Lucas llega a derecho y se encuentra allí con Carmen. Al hablar con ella, Carmen le dice que le va a explicar el concepto de asignación, pero antes Lucas tendrá que conseguirle una chocolatina. Carmen le da una moneda de un euro y se queda esperando en la puerta principal de derecho. Lucas tendrá que buscar una máquina de snacks que se encuentra en la puerta secundaria de derecho. Allí podrá comprar una chocolatina para Carmen.

Cuando le dé la chocolatina, Carmen le explicará la asignación y le explicará que antes tenía 100 céntimos (un euro) y ahora tiene 20 céntimos y una chocolatina. A continuación Carmen le hará un test sobre asignaciones a Lucas. Las respuestas de este test son:

Pregunta 1

x:= 10; y:= 5; z:= 3;

y:= y-z;

x:= x-y;

¿Cuánto vale x?

1. 15
2. 5
3. 8 CORRECTA

Pregunta 2

x:= 4.0; y:= 0.5;

x:= x * y;

¿Cuánto vale x?

1. 4.5
2. 2.0 CORRECTA
3. 5.0

Pregunta 3

var1:= 10; var2:= 20;

var1:= var2 * 2;

¿Cuánto vale var1?

1. 40 CORRECTA
2. 20
3. 30

Pregunta 4

x:= 10; y:= 15;

x:= x + x;

¿Cuánto vale x?

1. 10
2. 20 CORRECTA
3. 30

Pregunta 5

x:= 10.0; y:= 2.5; z:= 3.0;

x= (z * x) - y;

¿Cuánto vale x?

1. 27.5 CORRECTA
2. 24.0
3. 35.5

Pregunta 6

x:= 5; y:= 6; z:= 2;

x= (x - 1) * z;

¿Cuánto vale x?

1. 9
2. 8 CORRECTA
3. 10

Pregunta 7

var1:= 10; var2:= 3;

var1:= var1+ (1 –var2);

¿Cuánto vale var1?

1. 10
2. 8 CORRECTA
3. 12

Pregunta 8

var1:= 2; var2:= 6;

var1:= var1+ var2;

¿Cuánto vale var1?

1. 10
2. 8 CORRECTA
3. 16

Pregunta 9

x:= 3; y:= 2; z:= 4;

x:= (x - z) * y

¿Cuánto vale x?

1. -1
2. 2
3. -2 CORRECTA

A continuación Carmen le explicará la estructura IF y Lucas tendrá que realizar el *mini-juego* de la balanza. Para equilibrar la balanza, Lucas deberá elegir una combinación de cubos cuyo peso sume 12, igual que en el brazo izquierdo. Una vez conseguido, Carmen le propone un nuevo reto para comprobar que de verdad ha asimilado el concepto IF. Este reto aparecerá en forma de dos nuevos minijuegos, en el que le aparecerá una estructura IF-ELSE donde tendrá que seleccionar las opciones correctas para que el código tenga sentido (véase sección 4.1.7).

Tras superar estas pruebas, Lucas tiene una conversación con Carmen en la que se le explica lo que son los Ifs anidados y la sentencia CASE. Nuevamente tendrá que superar dos minijuegos para demostrar que ha comprendido la explicación (en la sección 4.18 se explica de forma detallada cómo superar estas pruebas).

Una vez superados, Lucas le dice a Carmen que Gonzalo no tenía el libro, y Carmen recuerda que el libro no se lo llegó a dar, y que lo escondió en una estatua cerca de allí, por la zona de Filosofía. Carmen manda a Lucas a buscarlo y quedan en verse en la puerta de la Facultad de Matemáticas, donde tiene que reunirse con su amiga Eva.

Para ir a la Facultad de Filosofía, donde parece que ahora se encuentra el libro, Lucas podrá coger el autobús, ir andando desde derecho o hacer uso de los minimapas.

En Filosofía hay una estatua con un libro en la mano. Es probable que ese libro sea el de Carmen, pero hay una verja que impide a Lucas acercarse a cogerlo.

Hay un chico en la puerta, que le dice a Lucas que tal vez el bedel le pueda abrir la puerta, pero que está encerrado en la facultad y suele salir poco. Para conseguir que salga habrá que romper el aparato de aire acondicionado que hay cerca del despacho, de manera que por el calor el bedel salga a la calle. Para romperlo Lucas tendrá que utilizar la fregona que encontró en Historia (cerca de Gonzalo) con el aparato de aire acondicionado.

Una vez que el bedel salga Lucas podrá hablar con él y pedirle que le abra la verja. El bedel accederá pero no le pondrá las cosas fáciles a Lucas. Lucas tiene que pulsar un botón y cruzar un laberinto corriendo a lo "Indiana Jones" antes de que la puerta del otro lado del laberinto se cierre. Si Lucas no llega a tiempo tendrá que volver al principio del laberinto a pulsar el botón para que la puerta se vuelva a abrir.

Una vez que lo consiga y tenga a mano el libro, Lucas no se atreverá a dejar la estatua sin libro, por lo que continuando con el estilo "Indiana Jones", tendrá que cambiar el libro por otro libro. Podría intentar cambiarlo por el "Gran libro de la programación", sin embargo es lo suficientemente importante como para no dejarlo. La única opción será buscar otro libro. Puede valer el libro que hay en el despacho de delegación de alumnos de informática, titulado "Alicia en el país de los bits". Si Lucas hace esto podrá coger el libro de Carmen e ir a Matemáticas, donde ha quedado con ella para dárselo.

Al llegar a la Facultad de Matemáticas Lucas se encontrará de nuevo con Carmen. Entonces se irán a una zona con siete contenedores de basura. Allí le explicará la estructura FOR y para ello le dará a Lucas unas cartulinas para que cumpla la sentencia for que hay escrita en un cartel sobre los cubos. Para esto Lucas deberá empezar por el primer cubo y poner la cartulina con el cero. A continuación,

poner una cartulina negra. Después poner el dos. Así hasta el final, poniendo una negra en los cubos impares y el número correspondiente en los pares.

Para explicarle el vector, Carmen le da una brocha a Lucas y le dice que vaya a por pintura. Si Lucas se mueve un poco por la zona de Matemáticas verá que hay una verja que está cerrada con un candado, detrás de la cual hay un cubo grande con pintura. Para abrir este candado necesitará algo, puesto que no le vale nada de lo que tiene hasta ahora. Lucas irá hacia la parte del edificio de Biología, y ahí encontrará una puerta que está abierta y que parece que conduce a un cuartito. Al entrar, Lucas se tropezará con algo y caerá al suelo golpeándose con varias cosas y rompiendo la linterna al caer. La luz está apagada, así que Lucas saldrá de la habitación, no sin antes recoger algo que se clavó al caerse al suelo. ¡Son unas llaves! Lucas intenta abrir el candado de la verja con las llaves y, sorprendentemente, funciona. Una vez dentro, Lucas utilizará la brocha con el cubo de basura e irá a la zona de los cubos.

Carmen le pedirá que pinte su nombre en los cubos de basura. Tras mostrarse un poco reticente, Lucas accede y pinta su nombre. Entonces Carmen le explica los vectores, y le pide que para continuar limpie los cubos antes de que antes los encuentre así. Para ello Lucas debe encontrar un trapo que hay en la zona del parque de las ciencias y usarlo con los cubos.

Entonces, Carmen le muestra otro cartel, y le pide a Lucas que introduzca las cartulinas de manera que se cumpla la condición del cartel. Para ello, Lucas deberá abrir el cubo número uno e introducir la cartulina con valor seis, abrir el tres e introducir el tres y abrir el cuatro e introducir el cero. Lucas podrá realizar esta tarea en cualquier orden, porque no afecta para nada en el vector.

Cuando haya introducido correctamente las cartulinas, Carmen comenzará a explicarle a Lucas lo que es un puntero, tras lo cual se inicia un nuevo minijuego. En esta prueba, aparecerá una pantalla con varias variables puntero definidas y Lucas deberá seleccionar en orden los tipos que crean que apuntan cada una de las variables. El orden deberá ser: int, real, char, bool, int.

Cuando Lucas supera la prueba y Carmen se convence de que ya tiene asumido lo que es un puntero, empieza a hablar sobre las estructuras de datos de memoria dinámica: listas, pilas y colas. En este caso Lucas no tendrá que superar ninguna prueba, ya que se trata de 3 pantallas meramente informativas (que sirven como una pequeña introducción a la memoria dinámica).

Tras esto, Lucas le dará a Carmen el libro que tanto ha buscado.

Al abrirlo, Carmen se muestra muy sorprendida y preocupada. Borja le ha robado los derechos del método de aprendizaje e intentará venderlos para hacerse rico, cuando ella lo que quería era ayudar a la gente que empieza a programar. Lucas le dice que no pueden consentirlo. Entonces Carmen le pide un último favor, su amiga Eva tiene algún problema y está en el ascensor que conduce a la Facultad de Historia. Carmen y Lucas se separan, Lucas va a ayudar a Eva y Carmen va en busca de Borja para impedirle salirse con la suya.

Lucas tendrá que ir a la zona del edificio nuevo, donde encontrará a Eva, la amiga de Carmen, que está con muletas.

Eva necesita ayuda, puesto que el ascensor está roto y necesita ir a la parte de abajo, pero sólo hay unas escaleras largas y empinadas por las que difícilmente

podría bajar alguien con muletas. Parece que la única solución será arreglar el ascensor de alguna manera.

Si Lucas mira el tablón de Derecho, encontrará un cartel de los estudiantes de físicas titulado *"¿Cómo funciona un ascensor?"* En el que aparecen la mitad de unas instrucciones para activar un ascensor y se anima a los estudiantes a pasar por físicas para hacer unos cursos. Si Lucas va a Físicas, mirando los carteles que hay cerca de la puerta encontrará otro cartel con la segunda parte de las instrucciones. De vuelta al ascensor, cerca de este encontrara un panel eléctrico que podrá abrir utilizando una palanca que hay en Químicas, apoyada en un coche.

Una vez que Lucas haya abierto el panel, podrá proceder a aplicar las instrucciones. Teniendo que activar los interruptores 1, 3, 4 y 5 y pulsando después el botón 1. Cuando funcione el ascensor Eva lo utilizará y bajará a donde está Lucas. En agradecimiento por haber arreglado el ascensor, Eva decide explicarle lo que son las funciones y los procedimientos en Programación. Se cargará un minijuego que consta de dos partes, por un lado la definición de los dos conceptos, y por el otro dos pequeñas pruebas en las que Lucas tiene que demostrar su valía.

Las pruebas consisten básicamente en seleccionar elementos en orden de una lista para colocarlos en un trozo de código de manera que funcione correctamente.

Una vez que le haya dado las gracias Eva recibirá un extraño mensaje de Carmen al móvil. Parece ser que Carmen está en problemas y pide ayuda, está en el pasadizo de Biológicas. Si Lucas ya ha intentado entrar alguna vez no habrá podido entrar ya que da bastante miedo, sin embargo, una vez más tendrá que afrontar sus miedos para ayudar a su amada.

Una vez que Lucas entra en el pasadizo de Biológicas se encuentra con Borja, que parece estar loco. Ha encerrado a Carmen y no la dejará salir a menos que Lucas supere una última prueba. Una especie de examen en el que tendrá que responder a todas las preguntas correctamente. No liberará a Carmen hasta que no lo consiga.

Preguntas:

- **Pregunta 1**

```
for x:=1 to 5 do
{
        v[x]:= x*2;
}
```

¿Cuánto vale v[3]+v[5]?

1. 16 (CORRECTA)
2. 14
3. 12

- **Pregunta 2**

```
x:=0;
while (1==1) do
{
        x:= x +4 ;
}
```

¿Cuál es el valor de x después de ejecutar este código?

1. 0
2. 4

3. nunca saldría de este bucle → bucle infinito (CORRECTA)

- **Pregunta 3**

x:=3; y:=6; z:=1;

if (x==4)

    y:= x+y+z ;

else

    y:= z*2;

¿Cuánto vale y después de ejecutar este programa?

1. 6
2. 9
3. 2 (CORRECTA)

- **Pregunta 4**

x:= true; y:= false; z:=0;

if ((x and y) == true)

    z:= 10;

¿Cuál es el valor de z después de ejecutar este código?

1. 10
2. 0 (CORRECTA)
3. Hay un error de tipos

Pregunta 5

x:=true; y:=false; z:=5;

if (!x) then

    y:= x+z;

else


```
        y:= z*2;
```

¿Cuánto vale y después de ejecutar este programa?

1. True
2. False
3. Hay un error de tipos (CORRECTA)

- **Pregunta 6**

```
x:=true; y:=false; z:=true;
if (!x) then
        z:= x or y or z;
else
        z:= x and y and z;
```

¿Cuánto vale y después de ejecutar este programa?

1. False (CORRECTA)
2. Hay un error de tipos
3. True

- **Pregunta 7**

```
dinero:= 100;
preciocaramelo:=10;
caramelos:=0;
while (dinero>0) do
{
        caramelos:=caramelos+1;
        dinero:=dinero-preciocaramelo;
}
```

¿Cuántos caramelos tendré después de ejecutar este código?

1. 11
2. 9
3. 10 (CORRECTA)

- Pregunta 8

```
dinero:= 0;
caramelos:=0;
preciocaramelo:=10;
for caramelos:=1 to 6 do
{
        dinero:=dinero+preciocaramelo;
}
```

¿Cuánto dinero me costarán 6 caramelos?

1. 50
2. 60 (CORRECTA)
3. 70

- **Pregunta 9**

```
x:= 1; y:= 2;
if (x+1 == y) then
        x := x + y;
else
        x:= y-x;
```

¿Cuánto vale x después de ejecutar este programa?

1. 1
2. 2
3. 3 CORRECTA

- **Pregunta 10**

v[3]= 'a'; v[2]='m';

v[1]= 'e'; v[0]='g';

¿Qué palabra contiene v?

1. ameg
2. game (CORRECTA)
3. ninguna

Una vez superada la prueba Borja tendrá que liberar a Carmen y escaparse, pero durante la huida se le caerán los papeles de la propiedad intelectual del método de aprendizaje, y por fin Lucas y Carmen podrán estar juntos.

6. Conceptos enseñados

Como ya hemos mencionado anteriormente, lo que perseguíamos era iniciar en el mundo de la programación a aquellas personas interesadas en ello, y hacerlo de una forma entretenida y práctica, como es mediante un juego.

Siendo nuestro objetivo instruir en la programación, los conceptos que hemos incluido son los conceptos básicos que toda persona ha de conocer a la hora de enfrentarse por primera vez ante un lenguaje de programación. Nociones básicas y necesarias de las que se vale cualquier lenguaje de programación.

Durante el juego, según se van superando pruebas vamos adquiriendo nuevos conocimientos sobre programación en un orden lógico. Dichos conocimientos pueden ser consultados en todo momento en el "Gran Libro de la Programación":

- Algoritmo
- Constantes y Variables
- Tipos de Datos
- Tipos Enumerados
- Asignaciones
- Expresiones
- Sentencia If
- Sentencia Case y Tipos Subrango
- Sentencia While
- Sentencia For
- Vectores o Arrays

- Punteros, Listas, Pilas y Colas
- Procedimientos y Funciones

Con la intención de no resultar repetitivos animamos al usuario a que consulte la sección 4.7, Libro de Programación, donde están tratados todos estos conceptos de la programación de la misma forma que se hace durante el juego por si necesita aclarar alguna duda.

Debemos aclarar que cada uno de ellos suele representar cada una de las fases del juego.

7. Objetivos cumplidos

Al finalizar el trabajo en el proyecto hace apenas un año, los integrantes del grupo responsable dejaron abierta la posibilidad a futuras ampliaciones con las siguientes palabras:

"creemos que también se proporciona una buena base para crear un proyecto mayor, en el que se añadan conceptos nuevos (como por ejemplo los procedimientos, funciones y punteros), y principalmente el gran reto sería ampliar el número de pruebas y ejercicios para reforzar los conceptos ya explicados. Teniendo en cuenta esto, y otros factores mejorables del software desarrollado, creemos que algunas de las tareas a desarrollar en un futuro serían:

- Crear tramas secundarias, paralelas a la principal, en las que se realicen
- nuevas pruebas y se traten nuevos conceptos.
- Mejorar la calidad de algunos aspectos gráficos y sprites.
- Añadir sonidos y doblaje de voces a los personajes.
- Mejorar el formato de los diálogos que contienen código para hacerlo más legible. "

A continuación, explicaremos cuáles de estos objetivos han sido cumplidos y cuáles no y, en este último caso, las razones por las que no ha sido posible.

1. Tramas secundarias con nuevas pruebas y nuevos conceptos:

   Aunque la trama de la aventura gráfica era difícilmente modificable (ello suponía cambiar todo el sistema de fases, añadir personajes a la historia…y en definitiva alterar su orden lógico), sí que se podían añadir nuevas pruebas

para explicar conceptos adicionales a los ya introducidos con anterioridad – véase sección 4.1. de este mismo documento – .

2. Mejorar la calidad de algunos aspectos gráficos y sprites:

   Creemos haber conseguido grandes avances en este aspecto, mediante el tratamiento digital de todas las imágenes correspondientes a habitaciones, personajes y animaciones, objetos, etc.

3. Añadir sonidos y voces de doblaje a los personajes:

   Por las características y limitaciones del AGS esto no ha podido ser implementado, aunque de todas formas consideramos que una aventura gráfica de este estilo funciona bien sin voces de doblaje en sus personajes.

   No obstante, una música ambiental que le aporte dinamismo a la historia sería un buen avance y por ello ha sido investigado sin éxito durante la ampliación del proyecto.

4. Mejorar el formato de diálogos para hacerlo más legible:

   De la misma manera que en el punto 3, estábamos sujetos a las limitaciones del AGS por lo que poca cosa podía hacerse en relación al formado de los diálogos. Lo que sí se ha hecho, sin embargo, es modificar la velocidad de alguno de ellos de manera que al usuario le dé tiempo a procesar toda la información que aparece por pantalla.

   Asimismo, en los casos en los que se mostraba en forma de diálogo información relativa a código de un programa, o simplemente alguna explicación sobre conceptos en los minijuegos, se han creado nuevas

habitaciones con dicha información de manera que se evita saturar al jugador con demasiadas líneas de diálogo.

Además de estos cuatro puntos mencionados aquí, que corresponden precisamente a los objetivos futuros que marcaron nuestros compañeros el año pasado, se han llevado a cabo varias mejoras más ya comentadas en las secciones 3.3.2 y 4. de este mismo documento.

8. Conclusiones

Para finalizar nuestra memoria, queremos mencionar las conclusiones obtenidas sobre el trabajo que hemos realizado a lo largo de este curso. Primero, hablaremos sobre las necesidades más importantes que nos han surgido. Concretamente, de dos:

- La primera, fue la de tener que organizar el trabajo en fases y planificar el tiempo dedicado a cada una. Esto, nos costó un poco, debido a que nunca nos habíamos enfrentado a un proyecto de semejantes características. Sin embargo, hay que decir, que los cálculos que hicimos fueron bastante acertados, ya que en ningún momento nos sentimos agobiados. Pudimos compatibilizar este trabajo con el resto de asignaturas sin problemas.
- La segunda, la de tener que sincronizar el trabajo hecho hasta el momento. Para esta parte, ya teníamos algo de experiencia gracias a asignaturas de cuarto, que requieren la sincronización de un grupo de personas más numeroso. La resolvimos creando un servidor FTP, un grupo de Google, escribiendo continuamente correos electrónicos y por supuesto, quedando los tres, para tratar los temas más importante (reparto de tareas, marcar objetivos para cada fase, resolver problemas surgidos,…).

Otro tema relevante, es lo que nos ha aportado este proyecto.

Nos hemos dado cuenta de lo complicado que es transmitir conceptos de programación que para nosotros son tan básicos. Nos llevó horas y horas pensar en

ejemplos sencillos, pero que expusieran todo lo que queríamos de forma clara. Después tuvimos que materializarlos usando Paint, lo que tiene su punto de dificultad, con respecto a saber expresar mediante un dibujo, la idea que quieres. Esto, ha desarrollado nuestra creatividad, y nuestra faceta de enseñar.

También, recurrimos a nuestra imaginación y a la experiencia que teníamos con aventuras gráficas, para pensar en todas las mejoras que se podían hacer de este videojuego. Queríamos convertirlo en un juego más entretenido, intuitivo y divertido. Lo que pretendemos con el trabajo realizado, es que el jugador no tenga que estrujarse el cerebro demasiado para conseguir objetos que le permitan continuar jugando. Sino, darle puntos al superar pruebas de programación, para que consiga pistas que le dejen seguir con el juego. Así, se centrará en superar esas pruebas, que es el objetivo principal de este proyecto: que alguien que no sabe nada de programación, aprenda los conceptos básicos.

Resumiendo, este proyecto nos ha demostrado que no todo se reduce a la implementación de código. Sino que esta parte en realidad es la última fase del proceso. Nos ha enseñado que primero hay que pensar con detenimiento en lo que se quiere hacer. En este caso, en cómo explicar los conceptos nuevos, en cómo mejorar los ya explicados, atendiendo a cada palabra y dibujo que utilizábamos para que no llevaran a confusiones. También, en la idea de hacer un mapa y minimapas, que permitieran navegar por las estancias del juego de una forma rápida; en la ayuda requerida en cada momento por el jugador, en dar la opción de jugar en inglés, en ponerle fases, en cambiar la estética para hacerlo más "amable"… Y una vez pensado todo esto, escribimos el código que lo haga posible.

Creemos que hemos logrado todos los objetivos propuestos y esperamos que este proyecto ayude a las personas que tengan curiosidad por la programación.

9. Bibliografía

→ Ayuda propia de AGS:

[1] - Ayuda interna de AGS versión 3.1.

→ Libros:

[2] - Joyanes Aguilar, L. (1997). *Turbo/Borland Pascal 7*. Madrid, Mc Graw Hill.

→ Páginas web:

[3] - Wikipedia, la enciclopedia Libre. Artículo sobre AGS. Consulta: octubre de 2008. Disponible en:
http://es.wikipedia.org/wiki/Adventure_Game_Studio

[4] - Adventure Game Studio Forums. Foros de programadores AGS. Consulta: noviembre de 2008. Disponible en:
http://www.bigbluecup.com/yabb/

[5] - Creación de un juego con AGS. Consulta: noviembre de 2008. Disponible en:
http://www.elchiguireliterario.com/2007/04/10/creando-juegos-de-aventura-con-adventure-game-studio-i/

[6] - AGS Resources. Página web con recursos (CEditor) para programación en AGS. Consulta: diciembre de 2008 Disponible en:
http://www.americangirlscouts.org/agsresources/

[7] - Adventure Game Studio Official Webpage. Página web oficial de AGS. Consulta: diciembre de 2008. Disponible en:
http://www.adventuregamestudio.co.uk/

[8] - Manual general de AGS. Consulta: diciembre de 2008. Disponible en:

http://desarrollomultimedia.es/web-multimedia/manual-adventure-game-studio.html

[9] - Tutorial de scripts. Consulta: diciembre de 2008. Disponible en: http://desarrollomultimedia.es/articulos/tutorial-de-scripting.html

→ Trabajos anteriores:

[10] - De Miguel A.; Gómez B., (2008), "Memoria de Sistemas Informáticos: Videojuego educativo sobre introducción a la programación", Facultad de Informática, Universidad Complutense de Madrid.

Autorización:

Los autores del proyecto: Margarita García Azpiazu, Cristina Nicolás Fúnez y Alberto Vaquero Curto, autorizamos a la Universidad Complutense de Madrid a difundir y utilizar con fines académicos y de investigación, no comerciales, la presente memoria, así como el código, el software y el resto de material que componen este proyecto.

Firmado en Madrid, a 3 de julio de 2009

Margarita García Azpiazu	Cristina Nicolás Fúnez	Alberto Vaquero Curto
DNI 78.937.867-G	DNI 70.166.413-Z	DNI 70.878.761-Y